An Artificial Society of Token Traders:
Computer Simulation of Life, Happiness, and Complexity in Trade

by

Robert L. Turknett, Jr., B.S.

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Architectural Studies

The University of Texas at Austin

December, 1998

An Artificial Society of Token Traders:
Computer Simulation of Life, Happiness, and Complexity in Trade

Approved by
Supervising Committee:

Dedication

In loving memory of my grandmother, Clio Norris, who taught me to count.

Table of Contents

Chapter 1: Introduction

Imagine a group of people living in the same city, in close proximity to one another. Each of these people is connected to the others through a network of friends, family, and acquaintances. Some of these relationships people rely on very little; others they could not live without. Over time, old relationships wither and new ones form to fill the void.

This dynamic network of relationships represents a hidden economy, not of money and objects but of smiles and promises, of encouragements and invitations, of admonishments and refusals. Here a flower is exchanged for a kiss, there silence pays back a forgotten anniversary. A police officer is awarded a badge for completing her training, and a mother makes lunch for a child without expectation of reciprocation. These everyday exchanges of approval, disapproval, confidence, security and so on are the stuff of life, and the participants are as dependent on these exchanges for their happiness as they are dependent on food and water for their survival.

In this thesis we present a model of an economy like the one described above. Our model is based on a theory of human needs originally developed by Abraham Maslow and extended by Michael Benedikt in *Value: Economics, Psychology, Life* [1999]. Benedikt proposes a set of needs common to all human beings, and introduces the idea of a psychological economy in which individuals exchange tokens that satisfy these needs. The central argument of the book is that increasing complexity-and-organization – a pattern that characterizes all evolutionary processes – also characterizes human life and everything we deem to have value. The intent of this thesis is to provide evidence that increasing complexity-and-organization leads to greater need-fulfillment and therefore greater happiness among individuals in a society.

In Chapter Two we describe two models of value proposed by Benedikt [1999]. The first model defines value in evolutionary terms. The mathematical concept of complexity-and-organization is introduced, and positive value is associated with its increase. The second model defines value in terms of a Maslovian understanding of human needs and their satisfaction through trade. The theory of a psychological economy is presented.

Chapter III introduces TokenTrade, an agent-based computer simulation of a psychological economy. The underlying model is described in detail, and the computer program is presented from a user's point of view. Technical notes are provided for those interested in modifying the system.

In Chapter IV we present the results of three experiments conducted with TokenTrade. The data from these experiments is explained through graphs and screen shots of various simulation runs.

Chapter V contains conclusions drawn from these experiments. Three hypotheses are presented as potential subjects for future work.

Background and Previous Work

Cellular Automata and Complexity

In general, the present thesis is an attempt to study how patterns of increasing complexity-and-organization arise from simple interactions between autonomous agents. The question of how complex behavior arises from the interaction of simple components is the major focus of cellular automata research [Wolfram 1986]. Cellular automata (CA) share many characteristics with TokenTrade, and it is likely that phenomena encountered in CA research may turn up in TokenTrade as well.

Cellular automata (CA) are a class of mathematical systems characterized by discreteness (in space, time, and state values), determinism, and local interaction. Finite state cells are ordered in an n-dimensional lattice, and these cells are updated synchronously according to a deterministic local transition function *f*. Each cell obeys the same transition function, and the value of each cell is restricted to a set of integers $Z_k = \{0,1,\ldots,k-1\}$. A cell's value at any time step is determined by a function of the values of the neighboring cells at the previous time step. The general form of a two-dimensional cellular automaton is given by

$$x_{i,j}^{t+1} = f(x_{i-r,j-r}^{t},\ldots,x_{i,j-r}^{t},\ldots,x_{i+r,j-r}^{t},x_{i-r,j}^{t},\ldots,x_{i,j}^{t},\ldots,x_{i+r,j}^{t},x_{i-r,j+r}^{t},\ldots,x_{i,j+r}^{t},\ldots,x_{i+r,j+r}^{t})$$

$$f: Z_k^{2r+1} \to Z_k,$$

where $x_{i,j}^t$ denotes the value of cell (i,j) at time t, f represents the transition function, and r is a non-negative integer specifying the radius of the rule.

A two-dimensional cellular automata is often graphically displayed as a grid, with each cell in the grid having a certain color indicating its current state. For a simple, binary site-valued cellular automata, these colors might be white and black, indicating an "on" or "off" state.

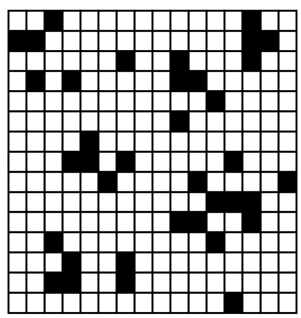


Figure 1.1: A two-dimensional, binary site-valued CA.

In recent years, cellular automata have arisen in popularity as a means for studying the behavior of various systems consisting of a large number of simple, identical, and locally connected components. Cellular automata have been used to model complex phenomena that arise in physical, chemical and biological systems [Canning and Droz 1990, Hartman and Tamayo 1990, Sieburg *et al.* 1990].

Much cellular automata research is focused on understanding the conditions under which non-random, non-periodic (or in our terms, complex-and-organized) behavior arises. Wolfram's [1984] scheme for classifying cellular automata on the basis of their dynamical behavior divides cellular automata into four classes:

Class I CA evolve to a fixed, homogeneous state.

Class II CA evolve to simple separated periodic structures.

Class III CA yield chaotic aperiodic patterns.

Class IV CA yield complex patterns of localized structures.

The most interesting CA are those that exhibit class IV behavior, and much research has gone into identifying their unique qualities [Li *et al*.1990, McIntosh 1990]. Langton [1990] has discovered a metric that he calls the λ parameter, which correlates directly with the amount of disorder observed. The λ parameter is simply the percentage of neighborhood states that do not lead to the quiescent state ("death"). By varying the λ parameter, we progress from CA exhibiting the maximum possible order to CA exhibiting the maximum possible disorder. The progression of behaviors as a function of λ is:

fixed-point $\rightarrow$ periodic $\rightarrow$ "complex" $\rightarrow$ chaotic,

or, in terms of the Wolfram classes

I $\rightarrow$ II $\rightarrow$ IV $\rightarrow$ III.

At intermediate values of λ, a phase transition is observed between periodic to chaotic dynamics, and behavior in the vicinity of this transition seems "complex," i.e. orderly and yet unpredictable.

Agent-Based Modeling and Artificial Societies

In this thesis we apply agent-based modeling techniques to study psychological exchange between individuals in a society. Agent-based modeling is useful when the object of study is a complex system containing many autonomous, heterogeneous, interacting agents. Although agent-based modeling has its roots in the theory of self-reproducing automata [von Neumann 1966], only in the past decade has large-scale agent-based modeling become possible due to recent advances in computing.

Although agent-based modeling has traditionally been applied to physical and biological systems, it is gaining popularity in the social sciences under the title of *artificial societies* [Epstein and Axtell 1996, Gilbert and Conte 1995]. Recent use of agent-based models in the social sciences includes the work of Albin and Foley [1990], Arifovic [1994], Arifovic and Eaton [1995], Arthur [1991, 1994], Axelrod [1993, 1995], Carley [1991], Danielson [1992, 1996], Epstein and Axtell [1996], Gilbert and Doran [1994], Gilbert and Conte [1995], Holland and Miller [1991], Kollman, Miller, and Page [1992, 1994], Marimon, McGrattan, and Sargent [1990], Marks [1992], Nagel and Rassmussen [1994], Tesfatsion [1995], and Vriend [1995]. The term "artificial society" refers to an agent-based model of a social system. An artificial society is composed of three basic elements: agents, an environment or space, and rules. Agents are the "people" of artificial societies, the environment is the medium in which they interact, and the rules govern their behavior [Epstein and Axtell 1996].

Epstein and Axtell have developed an artificial society called Sugarscape that shares many similarities with the TokenTrade model presented here. Sugarscape models a population of agents competing for food in an environment of scarce resources. The name Sugarscape refers to the two-dimensional landscape of "sugar" on which agents interact. This landscape is actually a cellular automata that models the growth of a generalized resource that agents require for energy. Agents are endowed with vision (ability to see sugar and other agents) and a metabolism (rate at which sugar is consumed). A movement rule governs their behavior. The simplest movement rule says "find the nearest sugar that you can see, go there and eat the sugar." In later chapters Epstein and Axtell add sophistications to the model including sex, combat, infection and trade between agents.

The most obvious difference between Sugarscape and the TokenTrade model that will be presented later is that of agent *movement*. In our model, agents will have a fixed location from which they cannot move. Because our focus will be solely on trade and not the migration dynamics of populations, we have chosen to isolate this variable in our model. However, there are many similarities between the models. Agent vision in Sugarscape corresponds roughly with neighborhood in our model, and our agents will possess a sort of metabolism which we call production/consumption rate. Trade in both models is based on the microeconomic principle that an exchange will only occur if at least one party is made better off by the exchange and neither party is made worse off. However, the way that trading partners are selected will differ. Neighbor selection in TokenTrade will be accomplished through a bidding process, while in Sugarscape, the neighbor that an agent trades with is determined solely by spatial proximity.

In designing the TokenTrade simulation, we have tried to follow the methodology set forth by Sites [1995] for building an exploratory simulation of a complex system:

<u>Typical Steps in Building an Exploratory Simulation of a Complex System</u>

- Simplify the problem as much as possible while keeping what is essential.
- Write program which (sic) simulates many components following simple rules with specified interactions and randomizing elements.
- Run program many times with different random number seeds, collecting data and statistics from the different runs.
- Attempt to understand how the simple rules gave rise to the observed behavior.
- Perform parameter changes and "lesions" on the program to locate the sources of behavior and the effects of different parameters.
- Simplify the simulation even further if possible, or add additional elements that were found to be necessary.

Our experience confirms the importance of simplifying the problem down to only the essential elements. Every bit of complexity that is added to a simulation makes it that much more difficult to track down the source of observed behavior. Complexity of the underlying model also leads to complexity of the implementation, meaning that errors and bugs in the code are more likely to occur, which may lead to incorrect conclusions if they go undetected. However, simplifying a model is often easier said than done. It is often difficult to know at the outset which elements are necessary to model a system and which elements are extraneous, so the model must be continually modified and simplified as what is important becomes clear through experimentation. Axelrod [1997] echoes these concerns and argues that replicating the results of simulation models, though rarely done, is essential if simulation is to become a legitimate research methodology in the social sciences.

Chapter 2: Two Models of Value

In this chapter we present two separate, but intimately related models that together form one cohesive theory of value, as originally developed by Benedikt [1999]. The first and fundamental model defines value in terms of an information-theoretic understanding of evolution. Evolution is a process by which the *complexity-and-organization* of a system increases, on average and over the course of time. Positive value is attributed to anything that leads a system in the direction of increasing complexity-and-organization. In the first section of this chapter we define complexity-and-organization in precise mathematical terms, and show how value is associated with its change.

The second model of value is an attempt to restate this first, fundamental model in terms of our own everyday, personal experience. Here, what is deemed valuable is that which satisfies our human psychological needs, that which makes us happy. In section two we introduce a Maslovian model of human needs and their satisfaction, and we describe a psychological economy based on the trade of tokens that fulfill these needs.

In the third section we suggest how these two models of value might relate to one another. As mentioned above, the information-theoretic understanding of value is considered fundamental. As such, the psychological-economic model can be viewed as a restatement of this fundamental model in human-centric terms. The goal of this thesis is to look for evidence that the psychological-economic model of value is, in fact, a restatement of the information-theoretic model.

Note that in our presentation of these two models of value, we do not attempt to persuade the reader of their validity, as this argument is beyond the scope of this thesis. The reader is referred to [Benedikt 1999] for philosophical arguments and empirical evidence that support the theory presented here.

I. THE EVOLUTIONARY MODEL

Information is measured in the transition from ignorance to knowledge. It is simply the difference, or change, in an observer's uncertainty, U, about a given situation *before* receiving some information, and his uncertainty *after* receiving the new information.

$$\boldsymbol{I} = U_{before} - U_{after},$$

where $\boldsymbol{I}$ means "amount of information gained." (2.1)

We define uncertainty as follows:

$$U = -\sum_{i=1}^{N} p(i) \log_2 p(i),$$

where $i = 1,2,3,\ldots N$ (2.2)

Each i represents one among N possible outcomes, and $p(i)$ is the probability of outcome i. Let us follow a simple example.

Imagine a situation in which we know that there are two possible outcomes. A coin flip is a good example. Before the coin lands, we are uncertain as to whether it will land heads up or tails up. The probability of the coin landing either way is “50-50” or 0.5, assuming our coin is unbiased.

$$
\begin{aligned}
U_{before} &= -[p(h)\log_2 p(h) + p(t)\log_2 p(t)] \\
&= -[0.5\log_2 0.5 + 0.5\log_2 0.5] \\
&= -[0.5(-1) + 0.5(-1)] \\
&= 1
\end{aligned}
$$

After it lands, we are certain of the way it landed, and the probability of the coin landing heads (or tails) up (whichever is actually the case) is 100%, or 1.0.

$$
\begin{aligned}
U_{after} &= -[p(h)\log_2 p(h) + p(t)\log_2 p(t)] \\
&= -[1\log_2 1 + 0\log_2 0] \\
&= -[1(0) + 0] \\
&= 0
\end{aligned}
$$

This gives us

$$I = U_{before} - U_{after} = 1 - 0 = 1\,\text{bit}$$

This transition from ignorance to knowledge, from “50-50” uncertainty to 100% certainty, emits one bit of information into the world, and specifically, one bit of information into the observer’s mind.

We can easily extend this to situations with more than two possible outcomes. An unbiased six-sided die roll, for instance, yields 2.585 bits of information:

$$
\begin{aligned}
U_{before} &= -\sum_{i=1}^{6} p(i)\log_2 p(i) \\
&= -[1/6\log_2 1/6 + 1/6\log_2 1/6 + \ldots 1/6\log_2 1/6] \\
&= 2.585 \text{ bits}
\end{aligned}
$$

$$
\begin{aligned}
&U_{after} = 0 \\
&I = U_{before} - U_{after} = 2.585 \text{ bits}
\end{aligned}
$$

This is, in fact, the maximum amount of information that we can receive about a six-possibility situation. This is true because we go from complete uncertainty to complete certainty about how the die will land. Uncertainty is greatest when all possibilities are equally likely to occur, as is the case with our unbiased die. In this case,

$$U_{before} = U_{\max} = \log_2 6.$$

In general,

$$U_{\max} = \log_2 N \tag{2.3}$$

If the die is biased in any way, and we know it, we receive less information from the die roll. Imagine, for instance, that a friend tips us off that the die is fixed, and that the probabilities are as follows:

$$p(\text{“1”}) = 0.1,\ p(\text{“2”}) = 0.4,\ p(\text{“3”}) = 0.1,\ p(\text{“4”}) = 0.1,\ p(\text{“5”}) = 0.2,\ p(\text{“6”}) = 0.1$$

We have gained

$$U_{before} = 2.585 \text{ bits}$$

$$\begin{aligned} U_{after} &= -\sum_{i=1}^{6} p(i)\log_2 p(i) \\ &= -[0.1\log_2 0.1 + 0.4\log_2 0.4 + 0.1\log_2 0.1 \\ &\quad + 0.1\log_2 0.1 + 0.2\log_2 0.2 + 0.1\log_2 0.1] \\ &= -[(-0.332) + (-0.529) + (-0.332) + (-0.332) + (-0.464) + (-0.332)] \\ &= 2.321 \text{ bits} \end{aligned}$$

$$I = U_{before} - U_{after} = 0.264 \text{ bits}$$

of knowledge from our friend. In all of the examples prior to this, U_{after} was equal to zero (certainty). This time, we are still left with some uncertainty, but we are less uncertain than we were before our friend tipped us off. Now if we roll the die, we have

$$I = U_{before} - U_{after} = 2.321 - 0 = 2.321 \text{ bits}$$

The act of rolling our biased die yields only 2.321 bits of information, 0.264 bits less than the unbiased die. This is because some of the information about the outcome was already known to us, *prior* to the rolling of the die − precisely 0.264 bits.[1]

Complexity

We can now define complexity in information-theoretic terms. Complexity, unlike information, is an intrinsic property of things-in-the-world, like their color or mass. The complexity of a thing does not depend on an observer, and is determined without regard to what we know or do not know about it. However, there is an intimate connection between uncertainty – call it apparent complexity – and complexity. The complexity of a system refers to *the amount of uncertainty the system produces in us when we are best informed about it*. We formulate complexity in terms of uncertainty, U, as follows:

Following Benedikt,

$$C \equiv U = -\sum_{i=1}^{N} p(i) \log p(i), \qquad \text{where } i = 1,2,3,...N, \tag{2.4}$$

$$C_{\max} = U_{\max} = \log_2 N , \tag{2.5}$$

---

[1] We should note that the measure of information presented here, I, is a slight modification of the one originally proposed by Shannon and Weaver [1949]. In their formulation of information, H, they assume that U_{after} is zero.

Whereas complexity refers to the "amount of randomness" in a system, organization, R, is a measure of the degree to which the system is constrained, or non-random. Just as information, $\boldsymbol{I}$, represents a reduction in uncertainty, organization, R, represents a reduction in complexity.

$$R = C_{\max} - C_{\text{actual}} \geq 0, \quad \text{or} \tag{2.5a}$$
$$= \mathrm{C}_{\text{potential}} - C_{\text{observed}} \geq 0 \tag{2.5b}$$

$C_{\max}$ and C_{actual} are objective, intrinsic properties of a system, while $C_{\text{potential}}$ and C_{observed} represent an observer's subjective knowledge about the system (which may be mistaken). While $C_{\text{actual}} <= C_{\max}$ and $C_{\text{observed}} <= C_{\text{actual}}$ for all circumstances, the relationship between $C_{\text{potential}}$ and $C_{\max}$, and C_{observed} and $C_{\text{potential}}$ depends upon the situation. When we have perfect knowledge of a system, $C_{\text{potential}} = C_{\max}$ and $C_{\text{observed}} = C_{\text{potential}}$. However, this is rarely the case, as we are prone to over- or under-estimate the actual as well as the potential complexity of a system. Since this state of imperfect knowledge is typical, we use the following hybridization of equations 2.5a and 2.5b to define organization[2], $\boldsymbol{R}$:

$$\boldsymbol{R} = C_{\text{potential}} - C_{\text{actual}} \tag{2.6c}$$

---

[2] For the purpose of explanation, this definition will do. The actual formulation is a tad messier: $\boldsymbol{R} = (\mathrm{C}_{\text{pot}}^2 - \mathrm{C}_{\text{act}}^2)^{0.5}$

Let us return to our previous example of a six-sided die. Now we are concerned with the *complexity* of the die (or more precisely, the complexity of the sequence of numbers that the die produces). We can easily compute $C_{\text{potential}}$:

$$C_{\text{potential}} = U_{\max} = \log_2 6 = 2.585 \text{ bits}$$

If the die is unbiased, then C_{actual} is equal to $C_{\text{potential}}$, since all possibilities are equally likely. In this case, we have:

$$\boldsymbol{R} = C_{\text{potential}} - C_{\text{actual}} = 2.585 - 2.585 = 0 \text{ bits}$$

This follows our intuition – an unbiased die is not organized in any way with regard to the sequence of numbers it produces. If we roll a six-sided die a thousand times, we should expect that each number will come up an equal number of times (between 166 and 167, to be exact). This is what we mean when we say that the die is random, or unbiased.

Now let us imagine that the die is biased, as we did earlier. $C_{\text{potential}}$ does not change, but C_{actual} is smaller than it was for the unbiased die.

$$C_{\text{potential}} = U_{\max} = \log_2 6 = 2.585 \text{ bits}$$

$$
\begin{aligned}
C_{\text{actual}} &= -\sum_{i=1}^{6} p(i)\log_2 p(i) \\
&= -[0.1\log_2 0.1 + 0.4\log_2 0.4 + 0.1\log_2 0.1 + \\
&\quad 0.1\log_2 0.1 + 0.2\log_2 0.2 + 0.1\log_2 0.1] \\
&= -[(-0.332) + (-0.529) + (-0.332) + (-0.332) + (-0.464) + (-0.332)] \\
&= 2.321\,\text{bits}
\end{aligned}
$$

$$\boldsymbol{R} = C_{\text{potential}} - C_{\text{actual}} = 0.264 \text{ bits}$$

We recognize these numbers from the earlier example, when we learned that our die was biased. From this we can see how the complexity of the die relates to our uncertainty about it. $C_{\text{potential}}$ is simply $U_{\max}$ from before, the maximum uncertainty we can have about the die. C_{actual} is equal to our uncertainty, U_{after}, after we learned that the die was biased. $\boldsymbol{R}$ is the information "contained" in the die, the information that we gained when we learned that the die was biased.

So far, we have been assuming that we have perfect knowledge of the die, but what about the case when our knowledge is incorrect, or at least, incomplete? Consider the following experiment. Paula is asked to listen as various numbers, between one and six, are read aloud from a list. She is instructed to listen carefully, and try to determine the pattern. After a certain period of time, she will be asked to show what she has learned by predicting what the next number will be.

Later, another woman, Kristen, is given the same instructions, but is told instead that the numbers in the list are between one and twelve. The list, however, is the same as before – it only contains numbers between one and six.

Unbeknownst to the subjects, the experimenter has generated the list by rolling a unbiased six-sided die, resulting in a list of numbers in random order. The actual complexity of the sequence, then, is $C_{\text{actual}} = \log_2 6 = 2.58 \text{ bits}$ for both of them.

After trying in vain to find a pattern in the sequence, Paula throws up her hands and reports that there is no order to the numbers. For Paula,

$$C_{\text{potential}} = C_{\text{actual}} = C_{\max} = \log_2 6 = 2.58 \text{ bits}, \text{ and}$$

$$\boldsymbol{R} = C_{\text{potential}} - C_{\text{actual}} = 0 \text{ bits.}$$

Since Kristen has incorrect knowledge about the list, $C_{\text{potential}}$ is not equal to $C_{\max}$. She has overestimated $C_{\max}$. After a period of time, Kristen comes to the conclusion that there are only six numbers (one through six), arranged in random order. In her case,

$$C_{\text{potential}} = \log_2 12 = 3.59 \text{ bits},$$
$$C_{\text{actual}} = \log_2 6 = 2.58 \text{ bits}$$

$$\boldsymbol{R} = C_{\text{potential}} - C_{\text{actual}} = 1.01 \text{ bits.}$$

Since Paula has perfect knowledge of $C_{\text{potential}}$, hers is the accurate measure of $\boldsymbol{R}$. After finishing the experiment, one might argue that Kristen knows that $C_{\text{potential}} = C_{\text{actual}} = 2.58$ bits, and $\boldsymbol{R} = 0$ bits, since she is now aware that the numbers seven through twelve never appear in the sequence. However, regardless of whether she revises her belief about $C_{\text{potential}}$, her uncertainty about the sequence has decreased. Paula, on the other hand, is still as uncertain as she was when the experiment began. In fact, we might say that Kristen comes out ahead psychologically, for she has the pleasure of gaining some information about the sequence. For her, $I = U_{\text{before}} - U_{\text{after}} = 3.59 - 2.58 = 1.01$ bits, while for Paula, $I = 2 - 2 = 0$ bits.

The Complexity of Fields

Our discussion of information and complexity thus far has focused on temporal sequences, linear successions of events in time. "Heads, heads, tails, heads,..." "5, 1, 4, 2, 3, 6, 3,..." "Red, blue, yellow, green, blue, blue,..." and the like. In many situations, however, we will want to determine the complexity of a system that is extended in space, rather than (or in addition to) time. To address this issue we turn to a discussion of the complexity of *fields* (or more precisely, the complexity of the behavior of temporally extended fields).

A field is a collection of symbols, states, etc. that is extended in space. Let us take the simplest case. Imagine a field containing four elements, arranged in a grid:

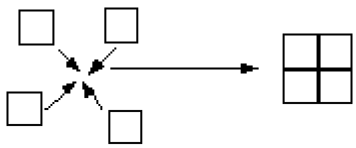


Figure 2.1: A field of four cells

Each element can be in one of two states, “white” or “gray.” Each cell, then, has potential complexity C_{pot} (cell) = $\log_2 (2) = 1$ bit. Each cell carries at most one bit of information. To find the maximum complexity of the field, we simply multiply by the number of cells, so that

$$C_{\text{pot}} = 4 \times \log_2 (2) = 4 \times 1 = 4 \text{ bits}$$

In general, the potential complexity of a field is given by

$$\begin{aligned} C_{\text{pot}} &= \log_2 N \\ &= \log_2 M^L \\ &= L \log_2 M \end{aligned} \tag{2.7}$$

where N is the number of possible field states, L is the number of elements, and M is the number of element states.

If our field was composed of elements capable of four states,

$C_{pot} = 4\log_2 4 = 8$ bits. If we extended this field to a third dimension, so that there were eight cells, we would have $C_{pot} = 8\log_2 4 = 16$ bits.

In order to compute C_{act}, we must have some idea about what states the individual elements in the field are likely to occupy in the next moment, just as we did with the coin, the die, and the screen. As we saw with these examples, if all states are equally probable, then $C_{act} = C_{pot}$. However, some states are more probable than others, $C_{act} < C_{pot}$, and we see organization in the behavior of the field.

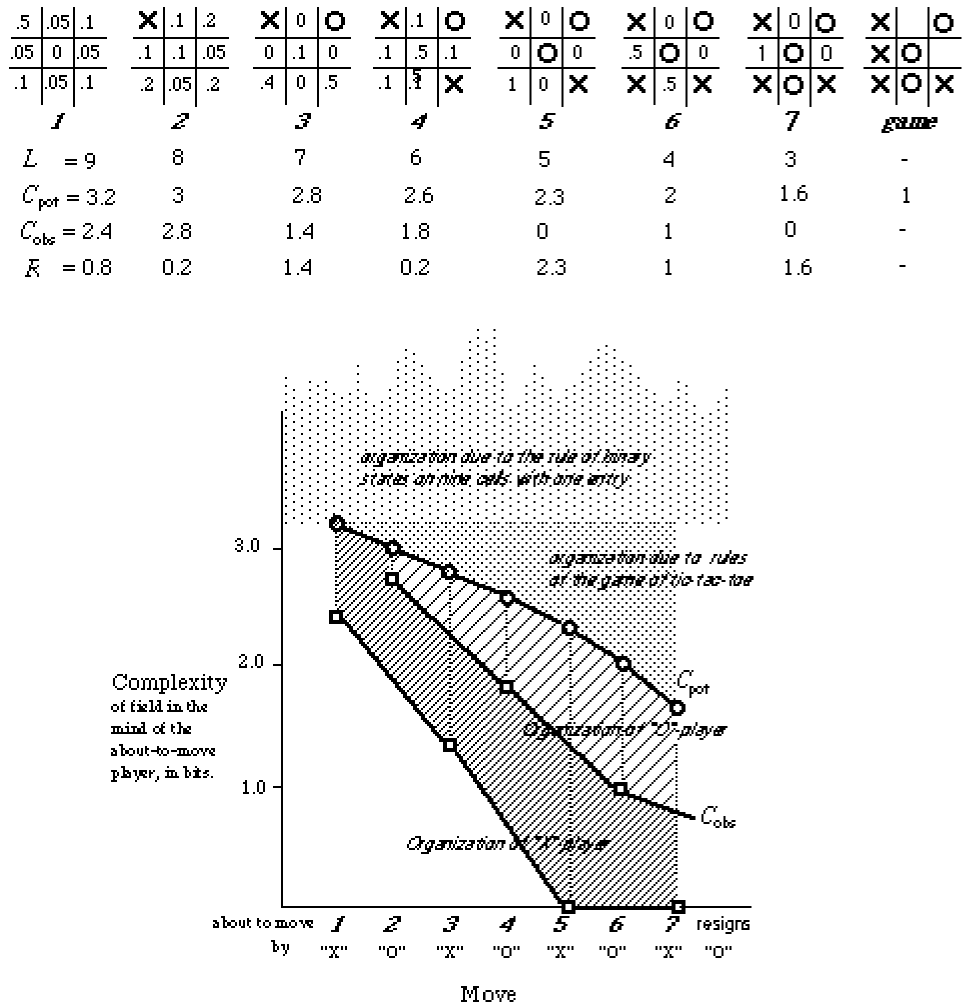


Figure 2.2: A typical game of tic-tac-toe (from Benedikt, 1999).

To illustrate this, let us imagine a game of tic-tac-toe. A tic-tac-toe board can be viewed as a two-dimensional field with nine elements, each having three possible states, “X,” “O,” and “empty.” For such a field, $C_{pot} = 9\log_2 3 = 72$ bits. However, in tic-tac-toe, C_{pot} is much smaller than this from the beginning, reflecting the organization imposed by the rules of the game. Let’s say that player

X starts the game. From X's point of view, there are only nine possible states of the field, corresponding to the nine possible moves available to X. Thus, for the first turn, C_{pot} = log N = $\log_2 9 = 3.2$ bits. Taking this number as C_{act}, we might say that the rules of the game bring R = $C_{pot} - C_{act} = 72 - 3.2 = 68.8$ bits of organization to the field.

At the start of the game, then, C_{pot} for player X is 3.2 bits. Now we compute C_{act} by determining the probability of each move, or field state. C_{act} represents the player's uncertainty about which cell in the field to mark next (i.e., which field state to choose). This of course depends on the knowledge that player X has about the game. In Figure 2.2 we follow a typical game of tic-tac-toe, estimating probabilities along the way. The conjectured probabilities represent two evenly matched players (elementary-age children, perhaps) with some experience playing the game, but not so much experience that the outcome of the game (a draw) is predetermined.

With each successive move, C_{pot} decreases as the number of possible moves, N, decreases by one. C_{act} also decreases as the possible moves become increasingly constrained by each player's situation.

Complexity-and-organization

Complexity-and-organization, denoted by the Greek letter Ω ("omega"), combines the two quantities, complexity, C, and organization, R, that we have previously introduced.

$$\Omega = \sqrt{(\text{complexity} \times \text{organization}} = [C_{\text{act}} \cdot R]^{0.5}, \tag{2.8a}$$

In our discussion of Ω, we use a slightly more elaborate measure of R:

$$\boldsymbol{R} = (C_{\text{pot}}{}^{2} - C_{\text{act}}{}^{2})^{0.5} \quad , \tag{2.9}$$

so that

$$\begin{aligned} \Omega &= [C_{\text{act}} \cdot R]^{0.5} \\ &= (C_{\text{act}}[C_{\text{pot}}{}^{2} - C_{\text{act}}{}^{2}]^{0.5})^{0.5} \\ &= (C_{\text{act}}{}^{2}[C_{\text{pot}}{}^{2} - C_{\text{act}}{}^{2}])^{0.25} \end{aligned} \tag{2.8b}$$

Our previous formulation of R is only slightly altered here, and this new function remains fundamentally similar to the equation for information, I. In our final formulation of Ω, we drop the subscript, and refer to C_{act} simply as C. Henceforth, "complexity" by itself shall refer to actual complexity, unless otherwise specified.

$$\Omega = (C^{2}[C_{\text{pot}}{}^{2} - C^{2}])^{0.25} \tag{2.10}$$

We can visualize the relationship of C, R and Ω by plotting complexity, C, against organization, R.

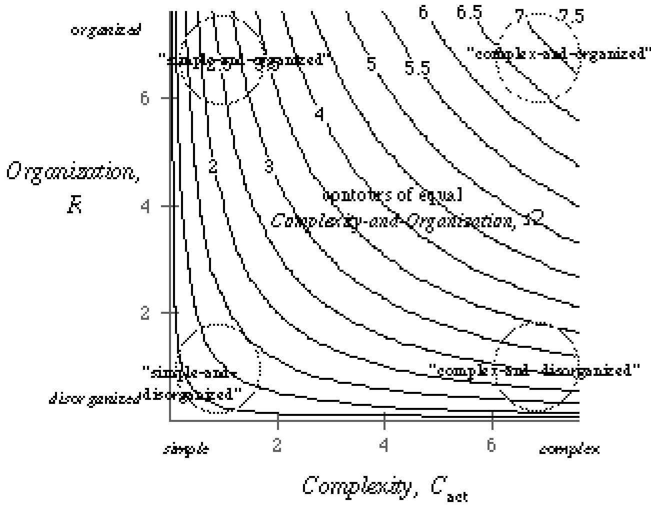


Figure 2.3: Graph of complexity, organization, and complexity-and-organization (from Benedikt, 1999).

The hyperbolic curves in the graph represent contours of equal Ω, such that anywhere along one of these contours, Ω remains the same. A system may be characterized depending on where it lies on this graph. On the left side of the graph, we have *simple* systems, on the right, *complex* ones. At the bottom of the graph reside *disorganized* (or "dis-ordered") systems, at the top, highly *organized* (or "ordered") ones. We can roughly describe a system as being situated at, or lying between, four extreme states: simple-and-organized, complex-and-organized, simple-and-disorganized, or complex-and-disorganized. Notice that systems with high Ω reside in or near the complex-and-organized extreme.

Potential complexity, C_{pot}, is not represented explicitly on this graph. However, it is easy to find C_{pot} at any point. Solving Equation 2.9, we get ${C_{\text{pot}}}^2 = R^2 + C^2$, and $C_{\text{pot}} = (R^2 + C^2)^{0.5}$. Thus, C_{pot} is the hypotenuse of a right triangle with sides of length R and C, according to the Pythagorean theorem. In Figure 2.4 we see that at any point on the graph, C_{pot} is simply the distance of the point from the origin.

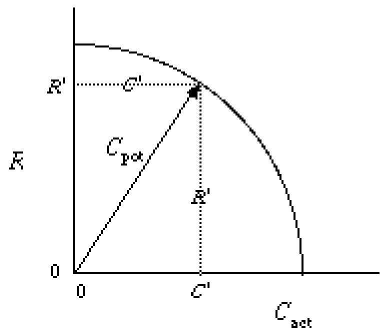


Figure 2.4: Finding potential complexity, C_{pot} (from Benedikt, 1999).

Figure 2.4 shows a radius sweeping out a curve along which C_{pot} does not change. Notice that as we move left on the curve, complexity, C, gets smaller while organization, R, grows. Moving to the right, R gets smaller while C grows. Also, we see that Ω is greatest for a given value of C_{pot} when R and C are equal.

This suggests a partitioning of Figure 2.3 into three regions, as shown in Figure 2.5. In the middle region, labeled "life," R ≈ C, and $C/ C_{pot} \approx 1/\sqrt{2}$. For a given C_{pot}, Ω will be greatest in this region. In the region labeled "rigidity," $R > C$, and $C/ C_{pot} < 1/\sqrt{2}$. Here, C/ C_{pot} approaches zero as C decreases. In the region labeled "chaos," $R < C$, and $C/ C_{pot} > 1/\sqrt{2}$. Here, C/ C_{pot} approaches one as C increases.

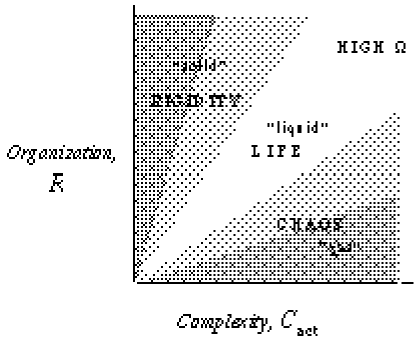


Figure 2.5: Three regions of the complexity-organization graph (from Benedikt, 1999).

Figure 2.6a combines the previous three figures into one graph. In addition to contours of equal complexity-and-organization, Ω, shown in Figure 2.3, we now see contours of equal potential complexity, C_{pot} , curving in the opposite direction. Furthermore, we have labeled the two regions "rigidity" and "chaos" as we did in Figure 2.5. A point along the C_{pot} contour C_{pot} = 7.0 is shown on the graph, at the midpoint of the contour where Ω is greatest. At this point, $C = R = \Omega = 4.95$. In general,

$$\Omega = \Omega_{max} \text{ when } \Omega = C = R = C_{pot} / \sqrt{2} \qquad (2.11)$$

Figure 2.6b shows the same contour, but this time Ω is shown on the Y-axis. From this we see that C does indeed have the same value as Ω_{max}.

In Figure 2.6c, C still occupies the X-axis, as in the previous graphs, but this time C_{pot} is shown along the Y-axis. A 45 degree line runs down the diagonal to the axis, along which C = C_{pot}. Between this line and the Y-axis are shown contours of equal Ω.

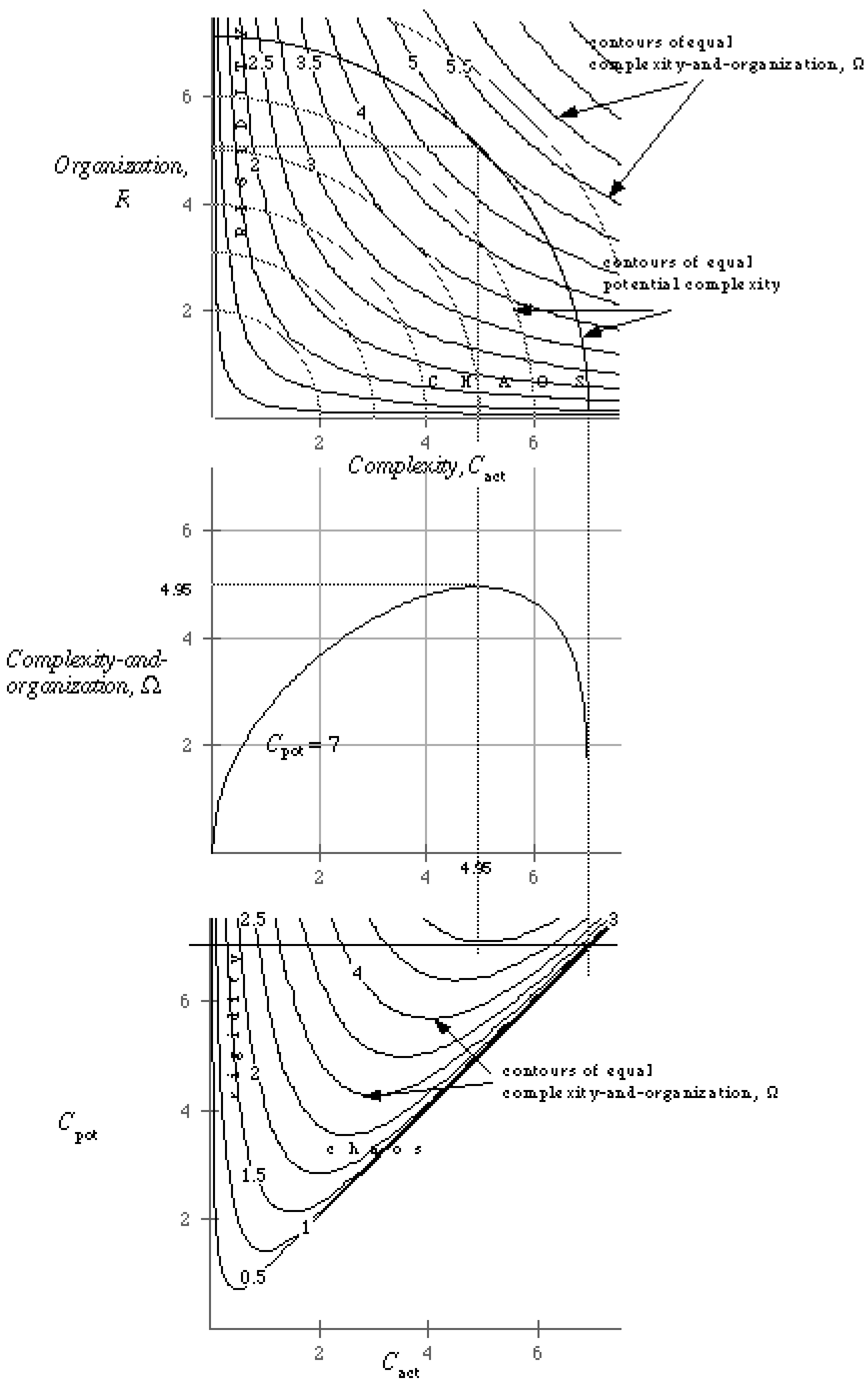


Figure 2.6a, b,and c: Three graphs of the relationship of Ω to C, C_{pot}, and R.

Evolution and Ω

Evolution, broadly defined, can be viewed as a process by which the Ω of a system increases, on average, over the course of time. The amount of complexity-and-organization, Ω, serves as a measure of the "evolvedness" or "lifefulness" of a system. Benedikt [1999] provides evidence from a number of empirical studies that support this use of the Ω-metric.

Perhaps the most persuasive of these is an analysis of gene sequences from various species showing the Ω-optimality of DNA at the scale of codons. DNA is made up of long sequences of nucleotides, adenine (A), guanine (G), cytosine (C) and thymine (T), grouped into triplets called codons. Each nucleotide triplet specifies an amino acid. The maximum number of different amino acids that DNA *could* specify with one codon is $4^3 = 64$, but in nature we find only 20. However, all 64 possible nucleotide combinations are found in DNA. Instead of omitting the unnecessary 44 combinations, three of these "extra" codons are used to encode a "stop" instruction, and the rest are redundant codes for the 20 amino acids, providing protection against error and noise. In this analysis, $C_{\text{pot}} = \log_2 4^3 = 3\log_2 4 = 6$ bits, and actual complexity, C, is determined by the frequency distribution of the 64 codons gathered from the "codon usage tables" for various species.

The results of this analysis reveal that all species lie along the contour of $\Omega = 4.242$ bits, which is equal to $\Omega_{\max} = C_{\text{pot}} / \sqrt{2} = 4.242$ bits. Thus, at the scale of codons, DNA is Ω-optimal.

Another example that is particularly relevant to the thesis presented here involves an analysis of a simple cellular automaton (CA) called LIFE [Gardner 1970, 1983]. Developed in 1974 by John Conway, LIFE is a 2-dimensional CA containing N binary state cells. The state of a cell at time step $t + 1$ is determined by the states of its eight adjacent neighbors, according to the following rules:

if exactly two neighboring cells are "on," stay in the current state.
if exactly three neighboring cells are "on," turn (or stay) "on."
in all other cases, turn (or stay) "off."

A starting configuration of "on" and "off" cells is specified, and the CA is allowed to run, updating with each time step *t*. Many initial configurations devolve quickly into an all-off state, or into an uninteresting periodic state. Some configurations, however, produce complex patterns that persist, travel, and evolve, and in some cases even reproduce, exhibiting some behaviors typically reserved to living organisms.

Configurations that exhibit the most interesting behavior have an "on" to "off" ratio of 1:5. Thus, for any cell, the probability that it is "on" is 0.2, and the probability that it is "off" is 0.8. This gives

$$
\begin{aligned}
C_{\text{pot}} &= 1 \\
C &= -\left[p(on)\log_2 p(off) + p(off)\log_2 p(off)\right] \\
&= -0.2\log_2 0.2 - 0.8\log_2 0.8 \\
&= 0.72 \text{ bits} \\
R &= (1^2 - 0.72^2)^{1/2} = 0.69 \text{ bits} \\
\Omega &= (0.72 \times 0.69)^{1/2} = 0.705 \text{ bits}
\end{aligned}
$$

This result is very close to optimal, $\Omega_{\max} = C_{\text{pot}} / \sqrt{2} = 0.707$ bits.

Near-optimal complexity-and-organization also turns up in the artificial society simulation Sugarscape, by Robert Axtell and Joshua Epstein. The most basic version of Sugarscape consists of a grid of cells, each of which may contain either an agent or some amount of "sugar" 0, 1, 2, 3, or 4. Agents require sugar to live, and they move around on the grid, or sugarscape in search of food. Their movement is based on what they are able to see in their neighborhood. Movement proceeds in rounds – with each round each agent in the population moves to the best source of sugar it sees.

Four qualities determine an agent's state: "vision," the number of neighboring cells seen 4, 8, 12, 16, 20, or 24; "metabolism," the units of sugar consumed each round 1, 2, 3, 4; "wealth," the amount of unconsumed sugar 0, 1, 2, … , 160; and location *(x,y)* on the sugarscape. Vision and metabolism are fixed, while wealth and location vary.

In Benedikt's preliminary analysis, Ω is optimal when vision = 8. An agent that is surrounded by sugar cells has 5^x possible neighborhood states to consider, where x is neighborhood size, or vision. Taking into account vision, metabolism, and wealth, the agent itself has only 6x4x161 = 3864 potential states

(Benedikt refers to these as "internal" states since they are not visible to an observer of the simulation). This gives

$$C_{\text{pot}} = \log_2 5^8 = 18.58$$

$$C = \log_2 3864 = 11.92$$

$$\Omega = 13.03 \approx \Omega_{\max} = C_{\text{pot}} / \sqrt{2} = 13.14$$

However, it is easy to find flaws with this analysis. Benedikt concedes that the analysis is incomplete, as it ignores the initial distribution of sugar, which is organized into two separate "hills" located in the northeast and southwest regions of the sugarscape, as well as the initial number of agents, which is 400 on a 2500 cell grid – a ratio close to the 1:5 ratio of "on"/"off" cells in *LIFE*. Furthermore, situations in which other agents are inside the agent's neighborhood are not taken into account.

As there are many different ways in which we might measure Ω in Sugarscape, it is important to be very specific about what it is that we are actually measuring. To state it precisely, what Benedikt has measured in the above analysis is the complexity-and-organization that results from the conversion (or compression) of neighboring sugar-cell state information into an agent state. Stating it in this manner reveals another problem with Benedikt's method: vision and metabolism are not variable, and thus are not part of the conversion of neighboring cell state information into the agent's state. Neighboring sugar-cell state information can be converted into wealth and position (which is absent in the above analysis) only, since these are the elements of an agent's state that are variable. This gives 161 x 2500 potential states, so that $C_{act} = \log_2 402500$. However, an agent never actually has the choice of 2500 (x,y) locations, since its position is constrained by its vision. Perhaps here we should draw a distinction between scales, as in the cellular automata example.

This analysis illustrates some of the difficulties one encounters when attempting to measure Ω in any system. Even with a relatively simple, well-understood system such as Sugarscape, measuring Ω becomes quite complicated. The more complicated the system, the more difficult it becomes to accurately assess Ω, as we shall see later when we attempt to measure Ω in our own TokenTrade simulation. Any measurement of Ω is always a partial view, and it is often difficult to determine the best place to insert the "probe" in order to capture the relevant potential and actual complexity. It is also important to distinguish between Ω that has been "designed-in" versus Ω that emerges from the simulation, by separating distinct scales of organization.

Value

Now that we have a way – at least in principle – to measure the "lifefulness" of a system, the definition of value is straightforward. Positive value is attributed to anything that increases the "lifefulness," or complexity-and-organization of a system. Value to a system is defined as a change in its complexity and organization:

$V = \Omega_{after} - \Omega_{before} = \Delta\Omega$

Any change in Ω, then, has value (positive or negative) to the system. Positive value is attributed to anything that "pushes" the evolution of a system in a positive direction, toward greater Ω. A system's location on the Ω graph determines which direction the system should move to increase Ω. Anything that moves the system in this direction has positive value. For instance, one system may need to increase C in order to increase Ω, while another system with the same C_{pot} may need to increase R by decreasing C in order to increase Ω. Imagine two such systems, A and B:

$C_{pot} = 3$

$C_A = 1.98$

$C_B = 2.44$

$\Omega_{max} = C = 3/\sqrt{2} = 2.12$

Thus, $C_A < \Omega_{max} < C_B$. In order to maximize Ω, system A must decrease C, while system B must decrease C.

Thus, what is valuable to one system might be of little or even negative value to another. In other words, the worth of things is relative. Also, note that any measurement of value requires a subject; value is always "value-to" a particular system that experiences a change in Ω. Furthermore, this subject may be a self-conscious, intentional being, doing everything in its power to increase Ω, just as it may be an inanimate system, such as a field of particles or an economy. The contention of this theory is that the very difference between animate and inanimate systems is their respective complexity-and-organization, and that the long-term survival of any of these systems depends on increasing Ω.

II. The Psychological-Economic Model

Benedikt's information-theoretic model of value provides an objective way to measure, for any system, the value of any change to that system. In this sense it is thought to be fundamental. However, it is unclear how this manifests itself in the daily affairs of human beings, where such assessments of value guide every choice that we make. We certainly do not believe ourselves to be making mathematical judgements about the complexity-and-organization of our lives when we choose to eat breakfast, play basketball, study pharmacology, or spend time with our family and friends. The psychological-economic model thus attempts to provide a model of value from a human point of view. Whereas the information-theoretic model of value provides a measure of value to *any* system, the psychological-economic (or psychoeconomic) model of value is concerned specifically with value to human beings.

In an extension of Abraham Maslow's original theory of personality, Benedikt proposes that all human beings have a similar set of psychological needs, and that happiness comes from the fulfillment of these needs. Needs arise in rough succession, so that higher needs appear only when our more basic needs are met. In this model, value (in the positive sense) inheres in those things that satisfy our needs. Benedikt extends Maslow's model to include the idea of a *psychological economy*, in which need-fulfilling *tokens* are exchanged between individuals and groups. In this economy, tokens have value relative to the needs of individuals, and individual happiness comes from the acquisition of needed tokens. Benedikt argues that this struggle to acquire tokens is the root of social evolution, the result of which is the increase in complexity and organization of human minds and human society.

The Stratigraphy of Needs

The stratigraphy of needs is based on Abraham Maslow's similarly titled hierarchy of basic needs, introduced in his book *Motivation and Personality*, which first appeared in 1954 [Maslow 1970]. Maslow, a psychologist, developed this theory of personality as an alternative to the Freudian and behaviorist theories that were popular in his time. Benedikt has revised this model to include six needs, one more than Maslow's original five, common to all human beings[3]:

[3] The claim that this applies to all human beings becomes more contentious at the higher strata in the stratigraphy. As we move upward, the needs are less directly biological and more socially rooted. As a result, there is probably more variation among cultures in the order in which these needs arise, or at least the order in which they are fulfilled. Benedikt notes that he is following a Western and modern structure.

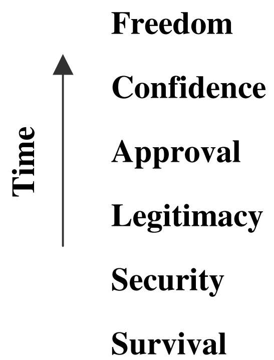


Figure 2.7: The Stratigraphy of Needs

An individual's satisfaction depends on the degree to which these needs are met. A perfectly satisfied person is one for whom all needs, from Survival to Freedom, are sufficiently fulfilled. *Happiness* is a feeling of progress that results as we become increasingly satisfied. In other words, happiness is the *rate of change* in satisfaction that we experience. We are most happy when satisfaction is increasing at a rapid pace. Likewise, unhappiness results from a rapid decrease in satisfaction.

Needs arise roughly in order from lowest to highest, beginning with the need for survival and approaching the need for freedom. The need for legitimacy, for example, asserts itself only when the need for security and survival are satisfied. The higher needs are considered to be more "evolved," and therefore more complex-and-organized, since they require a foundation of more basic needs for support. This is a point to which we will return later.

This picture of the needs as a strict hierarchy is too rigid, however. Lower needs are not abandoned with time, as this picture suggests, they simply become less urgent. Higher needs often co-exist with lower needs. Acknowledging this, Benedikt refers to his model as a "stratigraphy" rather than a "hierarchy" to suggest a layering of needs that is more flexible and organic. In this model, needs exist at overlapping strata rather than at separate levels. Adapted from Maslow, Figure 2.8 is offered as a more realistic picture of how needs arise and decline over time.

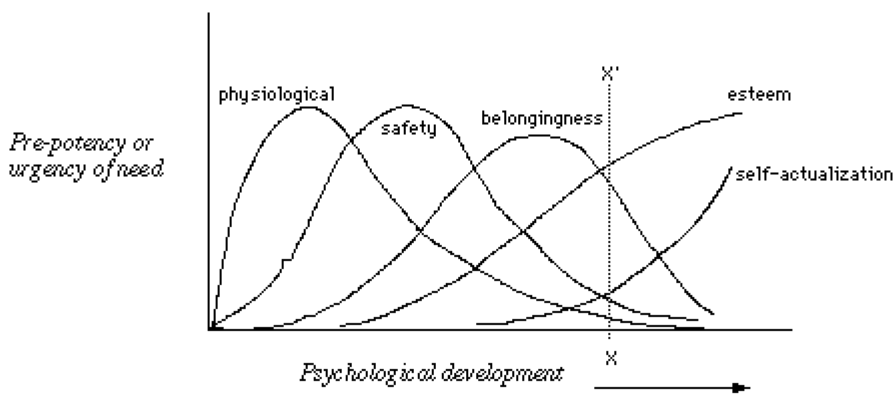


Figure 2.8: A picture of how the needs identified by Maslow succeed each other (adapted from Maslow).

This model imagines needs arising in strict order, from left to right. However, Benedikt argues that although this is a good picture of how *existing* needs succeed each other in strength, it does not account well for the emergence of new needs, which usually occurs during childhood. Thus, complementing the previous model of how needs succeed each other in strength is Figure 2.9, which shows how needs might emerge over time. In this model, needs emerge in the space between survival and freedom, becoming increasingly narrow and well defined and greater in number over time. In other words, the needs become increasingly specialized as the space between survival and freedom expands. This diversification of needs is thought to correspond with an increase in mental and behavioral complexity-and-organization of the individual.

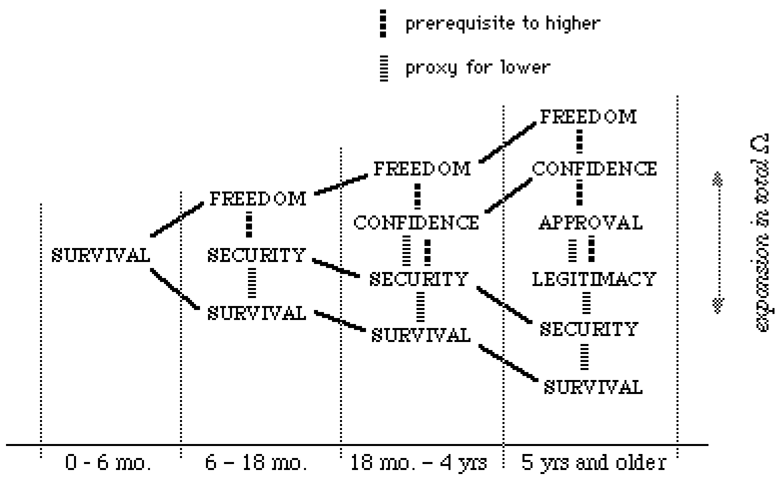


Figure 2.9: A model of how the needs might emerge over time (from Benedikt, 1999).

The expansion of needs is thought to occur through the introduction of *proxies* and *prerequisites* for already existing needs' satisfaction. Proxies decrease complexity, while prerequisites increase it. Proxies are substitutions for the satisfaction of (usually lower) needs, so that a proxy, like cash to gold, or gold to rice, acts as a stand-in for actual need fulfillment. Like cash, proxies serve as conveniences that increase the efficiency of trade, and their worth is likewise dependent on a collective agreement. Conversely, prerequisites delay trade by acting as new hurdles standing in the way of a higher need's satisfaction. Over time, proxies and prerequisites become established needs in and of themselves, thereby increasing the overall complexity-and-organization of the entire need structure.

The Psychological Economy

The model of individual needs that we have described thus far is only half of the picture. Benedikt proposes that needs are fulfilled by psychological goods called *tokens*, and that these goods are continually exchanged between people and groups in a psychological economy. Tokens fulfill specific psychological needs, and an individual's token preferences depend on his location on the stratigraphy of needs. This economy of tokens is thought to underlie the economy of material goods and services as traditionally understood.

Tokens

A (positive) token, in specific terms, is "a packet of information devised and offered by one party to another with the intention, on the part of the former, to provide or guarantee a measure of satisfaction of one or more of the latter's psychological needs, and generally in the expectation of reciprocation [Benedikt 1999, ch. 3]." Tokens appear at all levels on the stratigraphy. Figure 2.10 is a list of examples of various types of tokens, classified by the need that they fulfill:

Freedom tokens include invitations to, submissions of, admissions to; offers, consents, options, grants, access, tickets, releases, exemptions, immunities, waivers, absolutions, permissions, privileges; uncommitted time, unbudgeted money...

Confidence tokens include promises, secrets, entrustings, “inside information,” investments, loans, certifications, commissions, “cachet,” confirmations, sponsorships, scholarships, recommendations, encouragements, missions, votes, futures, stocks and bonds, checks...

Approval tokens include applause, smiles, compliments, kudos, votes, congratulations, grades, gifts, thanks, favors, blessings; expressions of pride, endorsements; dedications, autographs, kisses, declarations of love, honoraria...

Legitimacy tokens include names and namings (“en-title-ments”), legal identity (“papers”), charters, licenses, contracts, treaties, titles, prizes, medals, and awards, “station,” “office,” rank; stamps and seals, signatures, memberships and affiliations; (property) deeds, claims, obligations, justifications, rights, duties and responsibilities, proofs of provenance or lineage, salutes and other gestures of respect...

Security tokens include decrees, guarantees, sanctions, insurances, assurances, assessments of trustworthiness; “contacts,” guardianships, shelters, protection, patronage; seals and locks; tattoos, curses and caresses; alliances and allegiances; land, “inalienable” rights, responsibilities, and entitlements; predictability and regularity as such...

Survival tokens include life riskings, savings and sacrifices, life-critical knowledge; fasts, forfeitures, boycotts, oaths; scars; weapons; suicides and suicide threats, hostage takings, showing/getting “respect”...all demonstrative acts of violence, physical suffering/well-being, or nurturance; signals of sexual availability, indications of stocks of food, clothing, fuel, medicine, etc.

Figure 2.10: Examples of tokens [Benedikt 1999].

It is important to note that while objects may be the only physical evidence of a token exchange, tokens are psychological information, not physical objects. An apple presented to a teacher, for instance, represents a token of approval. However, it is not the apple itself that is the token, but rather the psychological information that transfers between student and teacher in the process of exchanging the apple. Because it is the psychological good that is relevant, the same object may be involved in very different token exchanges. The same apple given to a hungry child, for instance, once a token of approval, now becomes a token of survival. As the apple example illustrates, the context in which a token is offered determines its effect. A *gift* from a member of the mob, to give another example, is really an *obligation*. Thus, the same token may appear at different strata in the stratigraphy of needs in different situations.

Unlike material goods, tokens are immediately and completely consumed. Their effect of satisfying a need typically decays with time, after which the need must be addressed again. The rate at which this decay occurs varies with different types of tokens. A pat on the back is fleeting; a college diploma lasts for a lifetime.

Tokens can be positive or negative. Most of the examples in Figure 2.11 are put in positive terms, however, many of these examples have negative counterparts. A *frown*, for instance, is a negative approval token – the opposite of a *smile*. Furthermore, some tokens operate at more than one stratum simultaneously. A marriage license is one such "token bundle," as it transfers positive tokens of legitimacy and approval along with negative freedom tokens (in modern, liberal societies).

Finally we note that tokens do not always come from other people, as we often exchange tokens with ourselves. We reward ourselves with approval when we accomplish our goals, and punish ourselves with guilt when we behave badly. Of course, this internal trading is likely an internalization of external societal and familial social pressures. Furthermore, tokens may come from other animate and inanimate things in the world, such as a pet or a work of art. We surround ourselves with animals, plants and objects that provide us with constant sources of positive tokens.

Satisfaction and Happiness

Whether we are satisfied depends on our ability to acquire tokens that fulfill our needs, either from others or from ourselves. Each individual in the psychological economy is satisfied to a different degree depending on their success in acquiring need-satisfying tokens. Those whose needs are well taken care of are more satisfied than those whose needs are relatively unfulfilled. A perfectly satisfied person is one for whom all needs, from survival to freedom, are sufficiently fulfilled.

Happiness is the rate of change in satisfaction that an individual experiences. An individual is happiest when satisfaction is increasing at a rapid pace. Likewise, unhappiness results from a decrease in satisfaction. Being fired from a job, for example, causes a great deal of unhappiness due to the resulting loss of a major source of survival, security, legitimacy, approval, confidence, and freedom tokens.

Value

Value is defined in terms of need satisfaction. In the psychological model, value is attributed to any event, or anybody from which we gain positive tokens as well as the tokens themselves. Negative value is attributed to any event, or anybody that yields negative tokens as well as the tokens themselves. Simply put, we value whatever increases our satisfaction.

The value of any given token depends on the amount of positive or negative satisfaction that it provides. As we have seen, the value of a specific token is different to every person, since every person, although they have the same "default" needs, has those needs satisfied at a different level and sensitivity at a given time. The value of a token can therefore only be thought of in terms of "value to" some particular individual.

III. Ω AND HAPPINESS

In the previous two sections we presented two different models of value. To summarize:

<u>Information-theoretic model</u>

Value is attributed to that which increases the complexity-and-organization (Ω) of the system in question.

<u>Psychological model</u>

People ascribe value to that which increases their satisfaction.

It would seem at first glance that these two models of value have little in common, and are perhaps in opposition with one another. Where the information-theoretic model is domain-general and mathematically precise, the psychological model is human-centric and imprecise. However, it is the goal of this thesis to provide some evidence that the two models are actually two different lenses on the same phenomenon. We will argue that the information-theoretic model underlies the psychological model – that the empirical reality of the psychological model is a direct result of the operation of the principles described by the first model.

Note that the two models share an important parallelism: in both models value is determined in relation to a particular subject, and value is always associated with an increase, or change of state, rather than a state. In the first model, the subject may be any system, whereas in the psychological model the subject is always a particular or "average" human being. While in the information-theoretic model complexity-and-organization is the increasing quantity, in the psychological model it is the satisfaction of an individual.

We suggest that the innate human desire to increase satisfaction is tantamount to an innate desire to increase complexity-and-organization, a desire that is evolutionarily beneficial to the individual, and ultimately, to the society at-large. As humans and by extension perhaps all organisms increase their satisfaction, in so doing they increase the complexity-and-organization of their behavior. This complex-and-organized behavior of individuals is reflected in the structure of the society at large and their surrounding environment, so that we find increasing Ω in anything that they are in contact with.

The evolutionary account goes something like this. The human experience of satisfaction (biologically realized through endorphin production) has evolved for the purpose of rewarding life-sustaining behavior. Our complexity as organisms reflects the complexity of the environment in which we must survive. The fulfillment of our simple needs for survival (food, water, sleep, protection, sex) requires highly complex-and-organized social behavior in a world with scarce resources and competitive organisms, and has thus led to needs that are only indirectly related to survival. Climbing the stratigraphy of needs is, then, the process of increasing behavioral complexity-and-organization (Ω). It is in this climbing that we find happiness, and it is this increase in complexity-and-organization that we value.

We cannot hope to prove this ambitious hypothesis here. Rather, at best, we hope to provide evidence that warrants its further investigation. In the following chapter we describe a computer simulation designed to test the above hypothesis. Through experiments conducted with the simulation software, we hope to shed light on the relationship – if a relationship indeed exists – between complexity-and-organization and satisfaction, at least conceptually.

Chapter 3: The TokenTrade Simulation

In this chapter we present TokenTrade, a computer simulation based on the psychological-economic model of value presented in the previous chapter. TokenTrade models the exchange of tokens between independent agents in a psychological economy, and allows the experimenter to adjust various global and individual parameters in order to test their effect on population dynamics.

The first section of this chapter describes the TokenTrade simulation in detail. Section two presents the simulation from a user's point of view, with interface details and instructions for using the program. These first two sections provides sufficient information for the researcher who wants to run experiments with the program as is. The final section provides technical notes that may be relevant to those interested in extending this work or incorporating it in their own research.

THE TOKENTRADE SIMULATION -- DESCRIPTION

TokenTrade is a simulation of a psychological economy. In this economy, individual agents have needs that are fulfilled by tokens. To fulfill their needs, agents trade tokens with one another. Parameters such as neighborhood radius, maximum trade size, maximum initial endowment, fairness of trade and dissipation/production rate, control the behavior of individual agents. For example, some agents are token producers, while other agents are token consumers. Some agents are restricted to perfectly fair trades, while other agents are allowed to make quite unfair trades, and so forth. The dynamics of the population as a whole depend entirely on the parameter settings of individual agents.

The simulation takes place on a two-dimensional grid, where each grid cell represents an agent in the economy.[4] Cells visibly shrink and grow depending on their satisfaction level, and change color depending on their happiness. Happy cells are red; sad cells are blue. A comprehensive look at the interface will be presented later. For now, this brief visual description is enough to keep in mind as we look at the workings of the simulation.

Needs and Tokens

Agents trade tokens that fulfill needs. Each individual has three needs[5]:

Confidence/Freedom

Legitimacy/Approval

Survival/Security

---

[4] Henceforth, the words "agent" and "cell" will be used interchangeably.

[5] Note that the six needs from Benedikt's stratigraphy of needs have been reduced to three for the purposes of simplification. This should not significantly affect the behavior of the simulation.

Likewise, there are three types of tokens, one corresponding to each need. We will refer to them as x, y, and z tokens, where x tokens correspond to survival/security, y to legitimacy/approval, and z to confidence/freedom. Each cell is initially endowed with a random number of x, y, and z tokens, within a certain maximum set by the user. Tokens are represented by real numbers greater than zero, so that

$$x, y, z \geq 0$$

Satisfaction

A cell's total satisfaction depends upon the fulfillment of its needs. The lower, more basic needs dominate the higher needs, so the lower needs must be attended to first. For example, confidence/freedom (z) tokens are worthless to a cell if its need for survival/security (x) is not being met. Furthermore, the accumulation of tokens yields diminishing returns. Thus, the impact of x (or y, or z) tokens on a cell's satisfaction diminishes as the cell accumulates more and more x (or y, or z) tokens. This is modeled by the following equation for cell satisfaction:

$a = x/(1+x)$,

$b = y/(1+y)$,

$c = z/(1+z)$,

where $0 \leq a,b,c \leq 1$

$$0 \leq S = a(1 + b(1 + c)) \leq 3 \tag{3.1}$$

Happiness

A cell's happiness, H, is simply the change in satisfaction that occurs after a round of trading. Increases in satisfaction produce happiness, decreases in satisfaction produce unhappiness (sadness). The equation for happiness is simply

$$-3 \leq H = S_{\text{after}} - S_{\text{before}} \leq 3 \tag{3.2}$$

Trade

In order to fulfill their needs (and thereby increase their satisfaction), cells trade tokens with each other. Tokens are traded in a barter-style economy. In a single round of trading, each cell is given an opportunity to trade. Trading proceeds as follows:

A cell is chosen from the trading pool at random. At the beginning of a round, the pool contains all the cells in the universe.

The cell negotiates a possible trade with each of its neighbors.

Negotiation of a trade
A potential trade consists of two offers, one offer made by the cell to its neighbor, and one offer made by the neighbor to the cell. The amount of tokens x,y, and z a cell can give away is constrained by the cell's maximum trade size, which we will call W. A possible trade is computed by generating a random offer for both the cell and the neighbor it is "negotiating" with:

Cell makes random offer $[x,y,z]$, where $0 < x,y,z < W_{\text{cell}}$
Neighbor makes random offer $[x,y,z]$, where $0 < x,y,z < W_{\text{neighbor}}$

Of these possible trades, the trade that results in the greatest value is executed and the cell and its trading partner are removed from the trading pool.

OR

If there is no trade that results in an increase in happiness, the cell (but none of its neighbors) is removed from the trading pool.

Value of a trade
The value of a trade is simply the combined happiness that would result from the trade, subject to a fairness test. The degree to which a trade must be fair is controlled by a fairness parameter, f. The value of a trade is computed by the following equation:

$$V = H_{\text{cell}} + H_{\text{neighbor}} - f \, | \, H_{\text{cell}} - H_{\text{neighbor}} \, |, \tag{3.3}$$

where $0 \leq f \leq 1$.

Steps 1-3 repeat until the trading pool is empty. The display is updated, and a new round of trading begins.

Neighborhood Size

A cell may only trade with those cells that are in its neighborhood. The size of a cell's neighborhood is governed by its neighborhood radius, r, where $0 \leq r \leq 4$. A neighborhood radius of 1 means that a cell can trade with all of its surrounding eight neighbors. A radius of 2 means that a cell can trade with all eight immediate neighbors, plus the next sixteen surrounding neighbors, for a total of 24. Also note that the universe "wraps-around" at the edges, so that the north and south, east and west edges of the universe are connected.[6] Thus, a cell that is located on the western edge with $r = 1$ can trade with (three) cells located on the eastern edge of the universe.

Production/Consumption

Some cells "habitually" and continually produce tokens, while other cells habitually and continually consume them. This is controlled by a cell's dissipation/production rate, p. The dissipation/production rate governs the amount of tokens a cell produces or consumes per round. When p is positive, the cell produces tokens, when p is negative, the cell consumes tokens.

[6] Since the edges of the grid are "sewn" together, the universe would be more accurately represented as the surface of a torus.

THE TOKENTRADE SIMULATION – USER INTERFACE

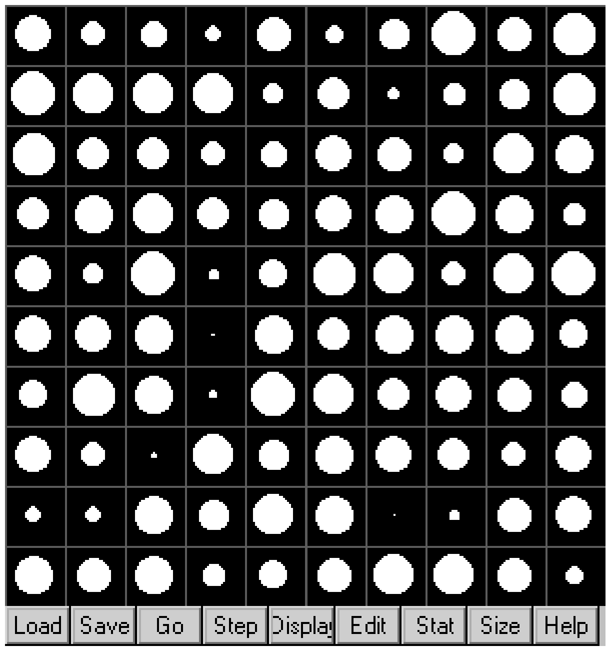


Figure 3.1: The main screen of the TokenTrade simulation (http://www.ar.utexas.edu/cadlab/turknett/tokentrade.html).

Upon loading the simulation on a Java 1.1 compliant web browser, the user is presented with the main screen of the program. The main screen consists of a square grid of cells, with a menu at the bottom. The menu contains seven buttons, each with a separate function, described below.

Load: Loads a population of cells. This only works if browser security settings are turned off.

Save: Saves a population of cells. This feature only works if browser security settings are turned off.

Go/Stop: Starts and stops the simulation

Step: Updates the universe, one step at a time. Each time the button is pressed, one round of trading occurs.

Display: Cycles between three different display modes.

Display Modes

Smiles indicate happiness, frowns indicate unhappiness. The fatness of the face represents satisfaction.

Red indicates happiness, blue indicates unhappiness. White indicates little change in satisfaction, or near-zero happiness. The size of a cell represents satisfaction. The brighter the color, the greater the change in satisfaction.

Same as display 2, with black indicating near-zero happiness.

Edit Cells: Allows the user to edit the parameter settings of selected cells. A group of cells must be selected with the mouse before pressing this button. There are five parameters that may be adusted.

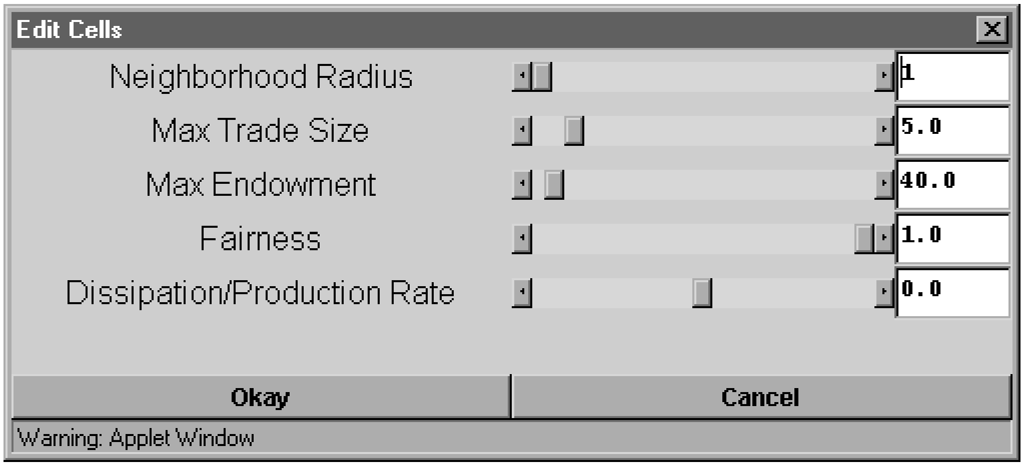


Figure 3.2: The "Edit Cells" window.

<u>Cell Parameters</u>

Neighborhood Radius (r). The radius of a cell's neighborhood. A cell's neighborhood consists of all the cells with which the cell can trade. $0 \le r \le 4$

Max Trade Size (W). The maximum number of tokens from each category x,y, and z that a cell can give away in a single trade. $0.0 \le W \le 50.0$

Max Endowment ($[x,y,z]$). The maximum number of tokens E from each category x,y, and z that with which a cell can be initially endowed. Endowment is randomly determined, so this sets an upper limit. $0.0 \le [x,y,z] \le 50.0$

Fairness (f). The degree to which trades initiated by a cell must be fair. A fair trade is one that results in the same amount of happiness for both cells. A trade that makes one cell happy while making another cell less happy is unfair. Note that a cell's fairness applies only to those trades it initiates. A cell with a fairness of 1.0 may still make an unfair trade with another cell, if the other cell is the one initiating the trade. $0.0 \leq f \leq 1.0$

Dissipation/Production Rate (p). The rate at which a cell consumes or produces tokens. Negative numbers are dissipation rates; positive numbers are production rates. A cell's tokens are decreased or increased by this percentage each round. When set to zero, the cell neither produces nor consumes tokens. Note that if a cell runs out of tokens, it can no longer produce tokens. $-0.1 \leq p \leq 0.1$

Stat: Brings up a window showing the statistics of the current run. Three statistics with corresponding graphs are shown in the window. This window updates every ten cycles. The "Save Statistics" button at the bottom of the window allows the user to save the statistics of the current run to an ASCII text file. However, this feature only works if the browser allows applets to write data to the client machine. The author has been able to accomplish this in Microsoft Internet Explorer by changing the browser's security settings for Java applets.

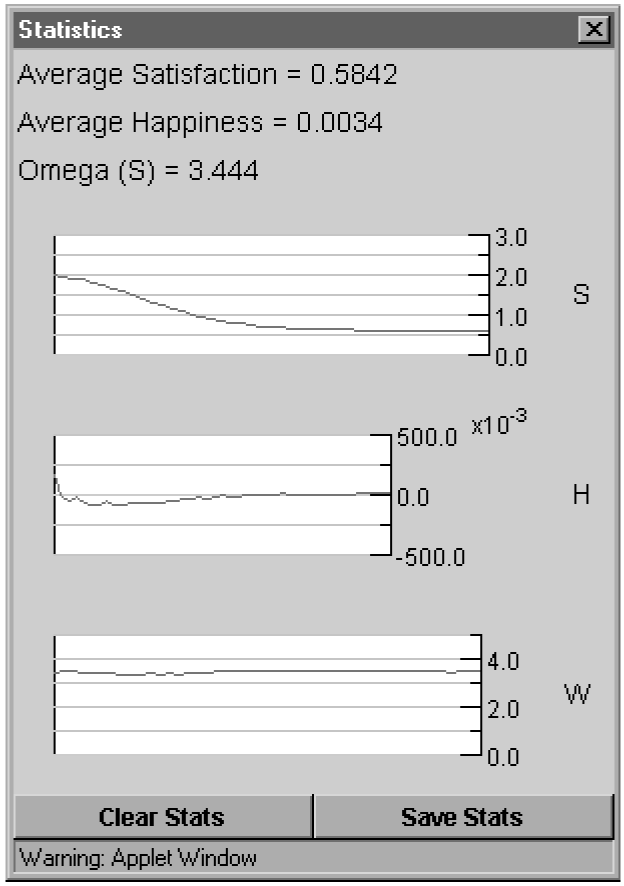


Figure 3.3: The "Universe Statistics" window.

Universe Statistics

Average Satisfaction (S). The average satisfaction of all the cells in the universe.

Average Happiness (H). The average happiness of all the cells in the universe.

Complexity-and-organization (Ω). This measures the complexity-and-organization of the distribution of cell satisfaction values across the whole universe. This is discussed in depth in chapter three.

Size: Changes the size of the universe. Size represents the number of cells across one dimension. Thus, a size of 10 represents a 10 x 10 universe. Note that larger universe sizes slow down the simulation, and require more memory to run.

Help: Displays information on how to use the program.

Selecting Cells

In order to change cell parameters, a cell or group of cells must be selected with the mouse. This is done in the usual way, by clicking on individual cells, and click-dragging to select a group of cells. Shift-selecting and Ctrl-selecting cells is also supported.

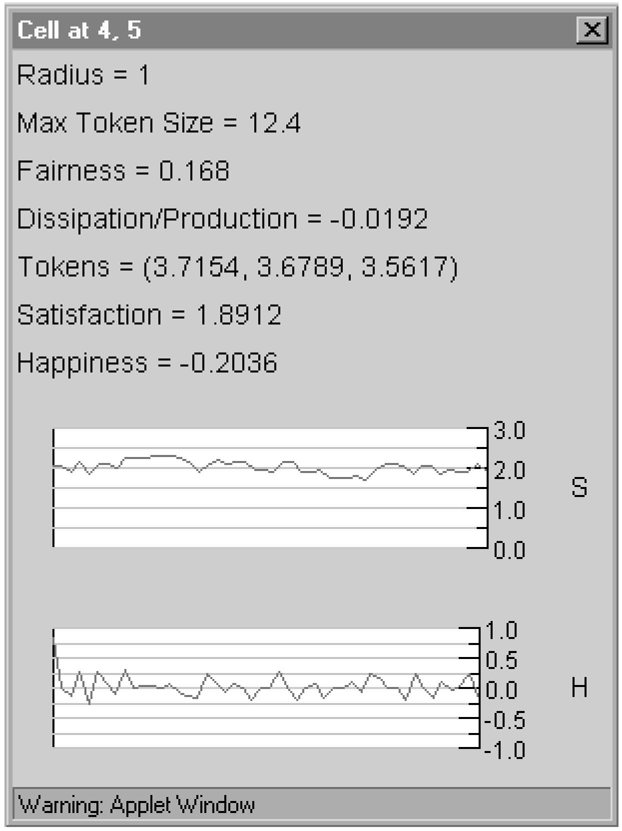


Figure 3.4: The "Cell Statistics" window.

Cell Statistics Window

Double-clicking on an individual cell brings up a "Cell Statistics" window. This window shows the current state of the cell, including cell parameters as well as current token, satisfaction, and happiness values. Satisfaction and happiness are shown on two separate graphs. This window is updated with each round.

THE TOKENTRADE SIMULATION – TECHNICAL NOTES

The TokenTrade simulation is written in Java version 1.1, an object-oriented programming language designed by Sun Microsystems, Inc. Java was chosen over C++ primarily because it offers platform independence, so that the simulation would run on a variety of operating systems, including Unix, Windows, and MacOS. Also, Java support is built into the two most popular Web browsers, Netscape Navigator and Microsoft Internet Explorer, which allows users to run the simulation remotely over the Internet by simply visiting the TokenTrade simulation web site (http://www.ar.utexas.edu/cadlab/turknett/tokentrade.html).

The drawbacks to using Java for this project were mostly due to its relative youth in the market. The most often cited criticism of Java is its sluggishness in comparison to C++, a point which is quite relevant here, since simulation programs are highly processor intensive. However, in the year since this project began, the speed at which the TokenTrade simulation runs has increased tenfold due to vast improvements in Java virtual machine (VM) implementation and Just-In-Time (JIT) compiler technology.[7]

Other problems encountered include poor and inconsistent VM support by various browsers, poor language support for graphical user-interface, and security restrictions for Java applets that prevent file input and output. Netscape's Java

[7] Based on qualitative observations using Microsoft Internet Explorer in Windows 95.

implementation still lags far behind Microsoft's in terms of speed and 1.1 feature support, and both browsers have been slow to adopt the new Java 1.1 standard. Furthermore, the same program will often behave differently depending on which browser is used, indicating that either the standard is not being adhered to, or that the standard is incomplete.

Graphical user-interface support in the current (1.1) version of Java is difficult to use and lacking in some basic functionality. For instance, simple slider widgets are not implemented in 1.1, and had to be programmed from scratch. Because of this lack of functionality, it was difficult to create a consistent, professional look-and-feel for the TokenTrade application.

Finally, security restrictions on Java applets prevent reading and writing files to the client machine, which hampers the usability of the simulation for serious experimentation because it is impossible to save the results of a trial run of the simulation. Using Internet Explorer, it is possible to get around these security restrictions by customizing the Java security settings.

The simulation could have been written as a standalone Java application instead of an applet, which would resolve the file I/O issue since applications do not have similar security restrictions. However we felt that the advantages of being able to access the simulation through a web browser outweighed these concerns. An applet will run on any machine with the latest version of Netscape Navigator or Internet Explorer, and only requires that the user go to the web page where the applet is located. An application, on the other hand, must first be downloaded by the user and then executed on the local machine. The user must have operating system support for Java 1.1 on his or her computer or the application will not execute. At the time of this writing, it is not possible to assume that this is the case for most users. Also, operating system support for Java seems to lag behind browser support, at least for the Windows platform. The end result of releasing the program as an application instead of an applet would be that fewer people would be able to use the program. However, it would not be difficult and would probably be worthwhile to create a corresponding application version with additional open and save features.

The good news is that Java seems to be improving rapidly on all of these fronts. As mentioned previously, the speed of Java virtual machines has greatly improved over the past year. Virtual machine manufacturers claim that their next releases will allow Java applications (and applets) to run at speeds nearly equivalent to native C applications, and native Java compilers will soon be available. The next version of Java (JDK 1.2) contains a replacement for the AWT, called Swing, which enables the creation of interfaces that are more integrated with the look-and-feel of other applications on the user's windowing environment, whatever that environment may be (Win95, MacOS, SunOS, etc).

Design decisions

The Java source code for the TokenTrade simulation is online at http://www.ar.utexas.edu/cadlab/turknett/tokentrade.html. The following section describes some of the reasoning behind several design decisions, and it may be helpful to those interested in understanding and/or modifying the original code.

Floating-point numbers and cell state

One of the defining characteristics of cellular automata (CA) is the notion of finite cell states. Each cell may be in one of several possible states. Often, cells have only two possible states, “on” and “off.” Taking all cell states together, the unique state of the CA at any moment in time may be determined. This allows for a straightforward mathematization of cellular automata systems, and is a major reason for their continued use as models of various real-world phenomena.

In TokenTrade, a cell’s state is represented by five variables. The first three variables represent the cell’s current stock of x,y, and z tokens. The remaining two variables represent the cell’s satisfaction and the cell’s happiness, respectively. Satisfaction and happiness of a cell are both computed directly from the token inventory $[x,y,z]$, so the token inventory is sufficient to determine the state of a cell.

Double floating-point numbers are used to represent tokens, satisfaction, and happiness. Floating-point numbers were used because the satisfaction function produces a real-valued result, and because floating-point numbers allow for greater variation and more realism in the simulation. However, the use of floating-point numbers throws into question the idea of finite state.

Floating-point numbers are intended to represent real numbered values, and real numbers are inherently non-finite. However, real numbers are theoretical entities, and their implementation on computers requires them to be represented as finite length byte arrays. Thus, cells are indeed finite state automata, but the number of states that each cell may occupy is many orders of magnitude larger than the number of states in typical cellular automata applications. In Java, double floating-point numbers consist of 64 bits, giving each cell 5 x 2^{64} potential states.[8]

Parallel vs. sequential updating

As explained above, the universe updates in rounds, meaning that each cell in the population must make a trade (or elect to pass) before the next round of trading can begin. This is intended to simulate simultaneous trading (many individuals trading at the same time). Because parallel computation is not possible with most computers, this method is commonly employed. Synchronous update is standard for cellular-automata, and is common in genetic algorithms and other agent-based simulations as well. An alternative to updating in rounds is randomly updating individual cells in the population (i.e., allow one cell to trade, then update its state and the state of its neighbor). One may also choose whether to require all cells to trade before a cell that has already traded may trade again. It

[8] Actually, since satisfaction, S, is a dependent variable of $[x,y,z]$, it is redundant information and should not be included in cell state. This is not the case with happiness, H,since it is a measure of change from the previous state. Thus, there are actually 4 x 2^{64} potential cell states.

seems likely that either of these options would have little effect on the behavior of the simulation, as long as trading cells are chosen randomly. In their Sugarscape simulation, Axtell and Epstein [1996] found no measurable difference between updating randomly and updating in parallel.

Wrap-around universe

The edges of the universe are connected in the simulation, forming a torus shape if the universe were displayed in three dimensions. This allows cells lying on the edge of the universe to trade with cells on the opposite edge. This is in keeping with the convention in cellular-automata literature. It would be very easy to modify the program so that the universe does not wrap-around, but it is unknown what affect this would have.

Chapter 4: Experiments and Results

In this chapter we present the results of several experiments performed with TokenTrade. These experiments provide evidence that there is a correlation between the mean happiness of a population of agents and the complexity-and-organization of the average satisfaction distribution of the population over time, as measured by Ω_S. We explore the nature of this correlation, and investigate how altering individual cell parameters affects the behavior of Ω_S with respect to satisfaction and happiness.

MEASURING COMPLEXITY-AND-ORGANIZATION IN TOKENTRADE

Ideally, we would like to devise a way to measure the complexity-and-organization of individual agent behavior over time. Thus far, no adequate measure has been developed and tested, so here we use a simpler method that measures the overall state of the entire population at a single point in time. We define a measure Ω_S , which represents the complexity-and-organization of the frequency distribution of satisfaction values at a given moment in time. Any one cell (agent) chosen at random thus has a probability of being in a certain S state that depends on this distribution over the population. C_{pot} is $\log_2 N$, where N is the number of possible satisfaction values. Since the range of satisfaction values, $0 < S < 3$, is continuous, we divide the range up evenly into 30 discrete sub-ranges, so that any given satisfaction value lies within a certain sub-range. Using this method of discretizing the range of satisfaction values, we calculate $C_{pot} = \log_2 30 =$ 4.91 bits. To compute C, we determine, for each sub-range, the proportion $0 \leq$ p(subrange) ≤ 1 of cells whose S is in that sub-range. This frequency distribution gives us the set of probabilities that the S of a randomly chosen cell in the population lies within particular sub-ranges. We then compute C according to equation 2.4, $C = -\sum_{i=1}^{N} p(i) \log p(i)$. Ω_S is then determined according to equation 2.8b.

Ω_S is essentially a measure of wealth stratification, where wealth is figured in terms of satisfaction. Ω_S will be highest when there are a number of "classes" of similar wealth, but not too many. If the population becomes too stratified, say, with one rich member and a throng of poor ones, Ω_S will be low, since organization is too high. Likewise, Ω_S will be low if everyone is rich or poor. On the other extreme, in a population where each member has a different degree of satisfaction, Ω_S will be low because complexity is too high.

One might argue here that using S in the computation of both Ω_S and average satisfaction ensures that there will be a correlation between the two, and that if a correlation is indeed discovered, we will only have proven what we have already implicitly assumed. However, a correlation between Ω_S and average satisfaction is by no means assured. Ω_S is a distributional measure of S and can vary almost independently of average satisfaction. When $S_{avg} = S_{max}$ or

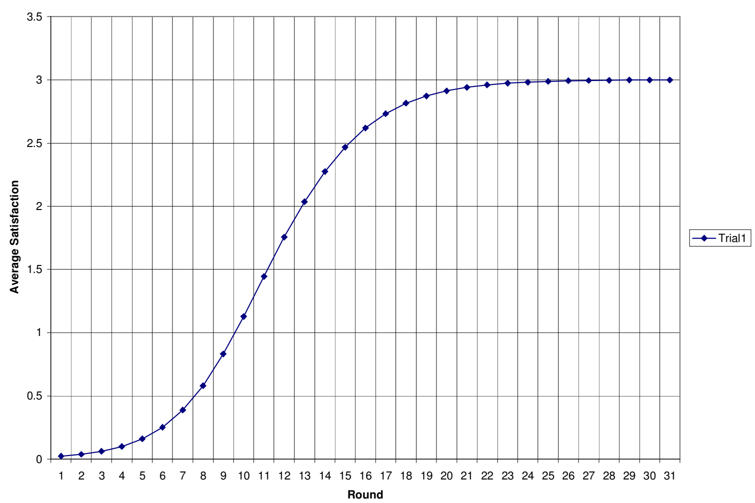
$S_{avg} = S_{min}$, $\Omega_S = 0$ because all the cells must be in the same state for either extreme to happen. But when S_{avg} is "middling," the value of Ω_S may vary greatly.

EXPERIMENT 1 – A POPULATION OF NON-TRADING PRODUCERS

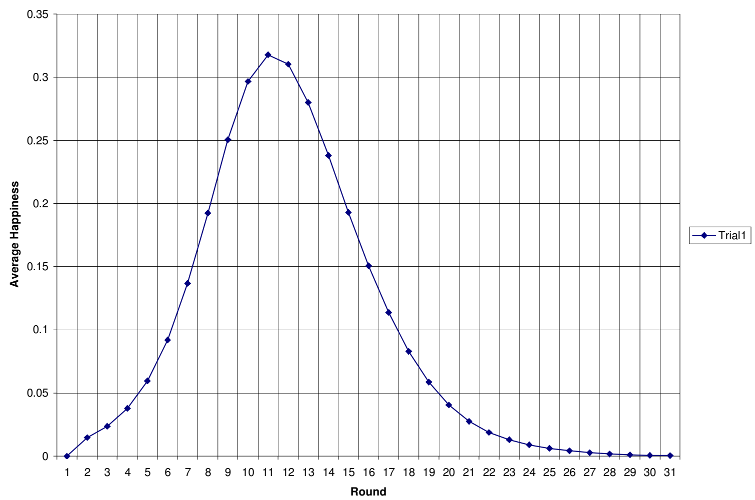
In the first experiment, we examine a baseline case in order to get an idea of how the simulation behaves with a uniform population when no trade is allowed. This will give us a better idea, when we do allow trading, of just what effect trade has on the behavior of the simulation. Each agent in the population of 100 cells is given the following attributes:

Radius = 1
Trade Size = 0.0
Endowment = 0.05
Fairness = 1.0
Production Rate = 0.05

Each agent begins with a small random endowment of tokens and produces additional tokens slowly. Since the trade size is zero, no trading occurs. An endowment of 0.05 means that each agent initially receives between 0.0 and 0.05 x,y, and z tokens, determined by a pseudo random-number generator. This results in an average initial satisfaction of around 0.02. Figure 4.1a shows average satisfaction in a sample run of this initial setup. The agents gain satisfaction as they produce tokens, accelerating upward to around 1.5. After this point, in round 11, the agents experience diminishing marginal returns from the production of additional tokens, and each subsequent increase in satisfaction is smaller than the previous one. Average satisfaction eventually reaches a plateau as it approaches $S_{max} = 3.0$. In Figure 4.1b we see that average happiness, H_{avg}, which is simply the first derivative or slope of S_{avg}, climbs initially until diminishing returns set in at round 11, where it peaks and then falls slowly to zero as the satisfaction curve levels out.[9]



[9] It is important to note that when we speak of a "round" here, we are actually referring to the moment when the universe statistics S_{avg}, H_{avg}, and Ω_S are updated, which happens after every ten rounds of trading. If we were to graph one data point for every round of trading, the results would look much the same, although small local variations might turn up that are smoothed out in the current graphs.



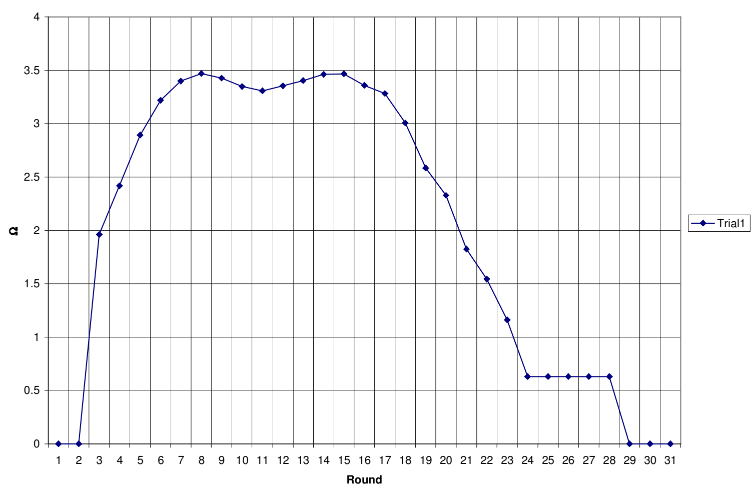


Figure 4.1a, b and c: Average satisfaction, average happiness, and Ω_S over time for a population of non-trading producers.

Figure 4.1c shows the behavior of Ω_S over time. Starting from zero, Ω_S climbs upward and decelerates to a peak just under 3.5, dips slightly, peaks around 3.5 again, and then descends back to zero. We find two maxima, one at round 8 and the other in round 15, with a local minimum at round 11. The curve is almost bilaterally symmetrical, like H_{avg}, with a midpoint at the local minimum in round 11. The shape of the curve seems to suggest a limit around 3.5, corresponding with the two maxima. Indeed, there is a limit here, corresponding with Ω_{max}. Using equation 1.11, we find $\Omega_{max} = C_{pot} / \sqrt{2} = \log_2 30/\sqrt{2} = 3.47$. Thus, at both maxima, Ω_S is optimal.

The behavior of Ω_S makes sense if we look at R and C.[10] In the beginning, R_S is very high because all agents have near zero satisfaction. As agents produce tokens, they gain satisfaction at different rates, since each cell begins with a slightly different endowment, and amount of tokens produced is a percentage of current tokens. Thus, initial differences in satisfaction values between agents are amplified and satisfaction values become more widely distributed across the range of possible values, causing a decrease in R_S and an increase in C_S. Ω_S is optimal when $R_S \approx C_S$ in round 8. When average satisfaction reaches the midpoint at 1.5 (round 11), happiness is greatest, and C_S is maximized. The local minimum at round 11 in the graph of Ω_S corresponds exactly with the peak, H_{max} , in the graph of H_{avg}. After this point, C_S begins to decline and R_S increases again. In round 15, $R_S \approx C_S$ again and Ω_S is maximized once more before declining to zero as R_S takes over.

There is clearly a direct correlation in this case between C_S and H_{avg}. As H_{avg} increases, there is greater variation of satisfaction values across the population, and therefore C_S increases as well. Furthermore, an analysis of the change in H_{avg} over time shown in figure 4.1d reveals a connection between the

[10] Unfortunately, we do not have a graph of R and C over time because the program does not currently output this data.

extrema in figure 4.1c and the inflection points in figure 4.1b. The maximum, minimum, and zero crossings in figure 4.1d correspond to inflection points in figure 4.1c. From this graph we see that the inflection points in round 8 and round 15 correspond to the maxima in figure 4.1b. Likewise, the zero crossing in round 15 corresponds to the local minimum in figure 4.1b. Finally, we notice that whenever H_{avg} is accelerating – upward in rounds 3-9 and downward in rounds 12-15 – Ω_S increases, and whenever H_{avg} is decelerating, Ω_S decreases.

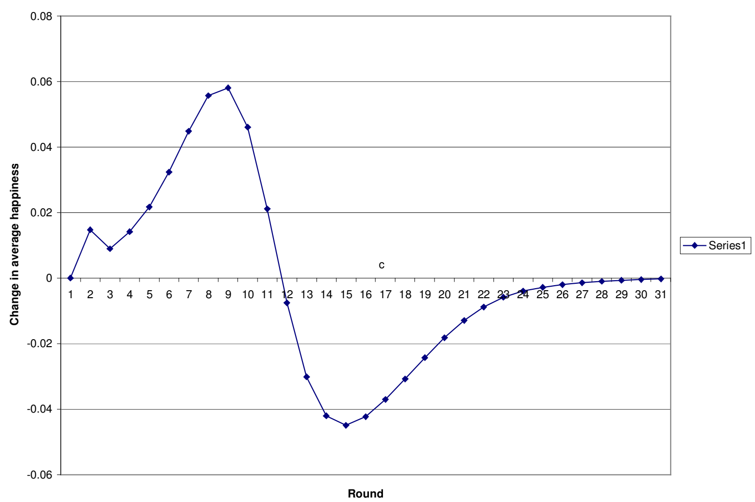


Figure 4.1d: Change in happiness over time for a population of non-trading producers.

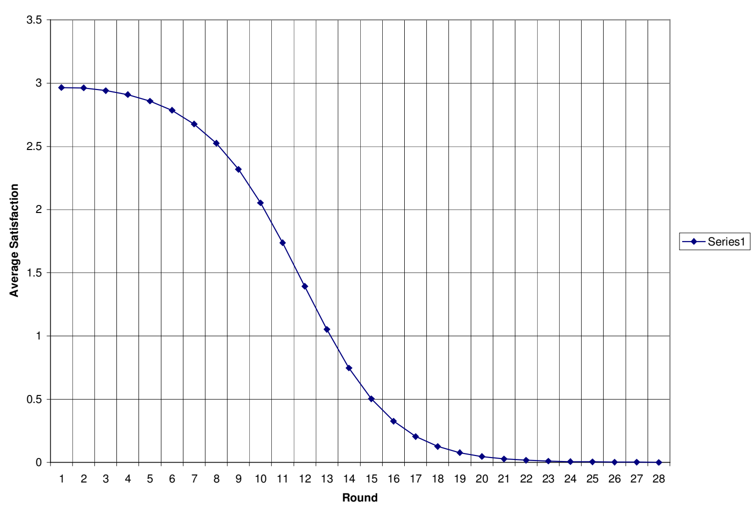
Reproducing this experiment with the same cell parameters yields nearly identical results time after time, with only slight differences due to the randomness of the initial endowment.

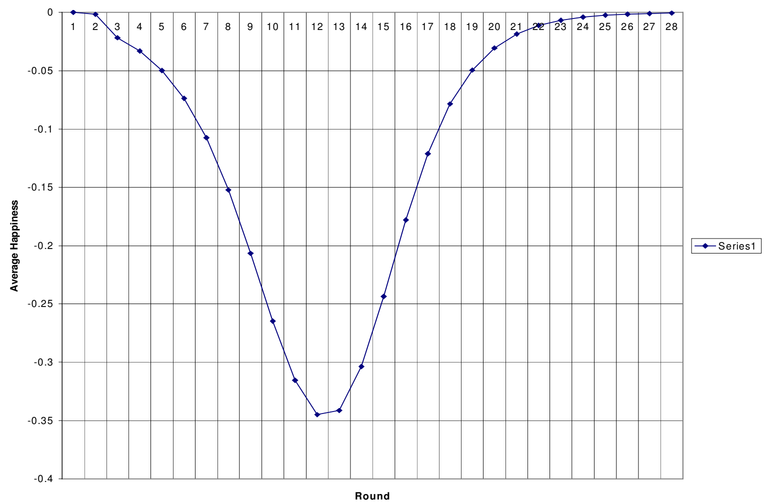
EXPERIMENT 2 – A POPULATION OF NON-TRADING CONSUMERS

In this experiment, we reverse the previous experiment by starting with a population of highly satisfied agents that slowly consume tokens until they have none left. Each agent in the population begins with the following attributes (again population size is 100 agents):

Radius = 1
Trade Size = 0.0
Endowment = 1000.0
Fairness = 1.0
Production Rate = -0.05

The picture here is nearly the exact inverse of experiment 1. Graphs of average satisfaction, average happiness, and Ω_S are shown in figures 4.2a-4.2c. Figure 4.2d shows the change in H_{avg} over time.





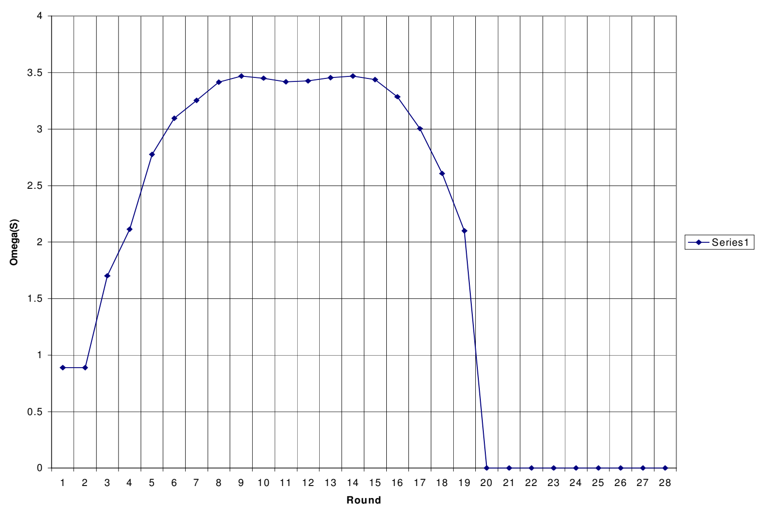


Figure 4.2a, 4.2b, and 4.2c: Average satisfaction, average happiness, and Ω_S over time for a population of non-trading consumers.

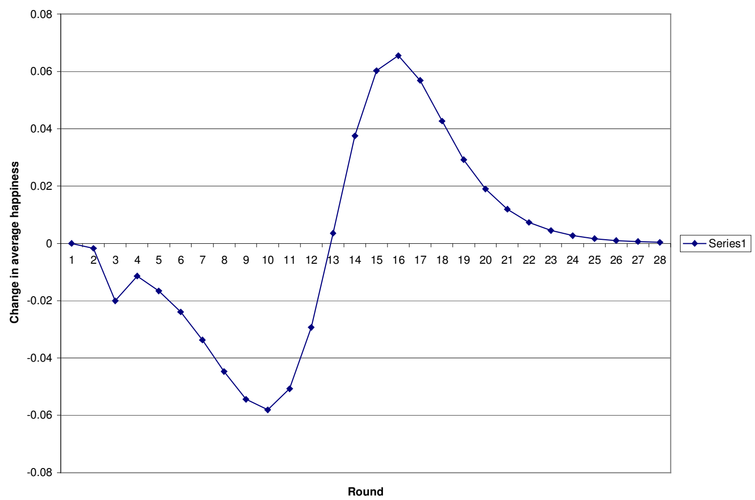


Figure 4.2d: Change in happiness over time for a population of non-trading consumers.

Once again we find that the inflection points in the graph of average happiness correspond with the extrema in the graph of Ω_S, and that increases in Ω_S occur in those rounds where the value of H_{avg} is accelerating, upward or downward. Both maxima in the graph of Ω_S are optimal at $\Omega_S \approx \Omega_{max}$. However, in this experiment the relationship between H_{avg} and C_S is inverted. As C_S reaches its peak, H_{avg} is at its lowest point. This suggests that C_S is not directly proportional to H_{avg}, but is instead directly proportional to the absolute value of H_{avg}. It is the magnitude of change in H_{avg}, not the direction that is important here.

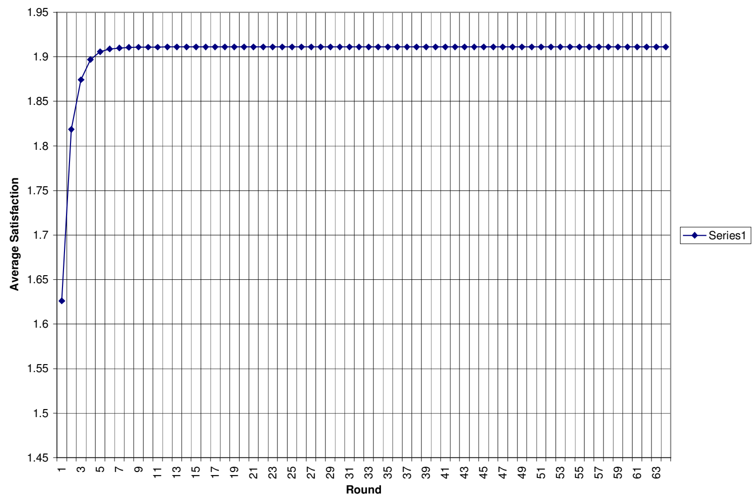
EXPERIMENT 3 – A POPULATION OF TRADING AGENTS

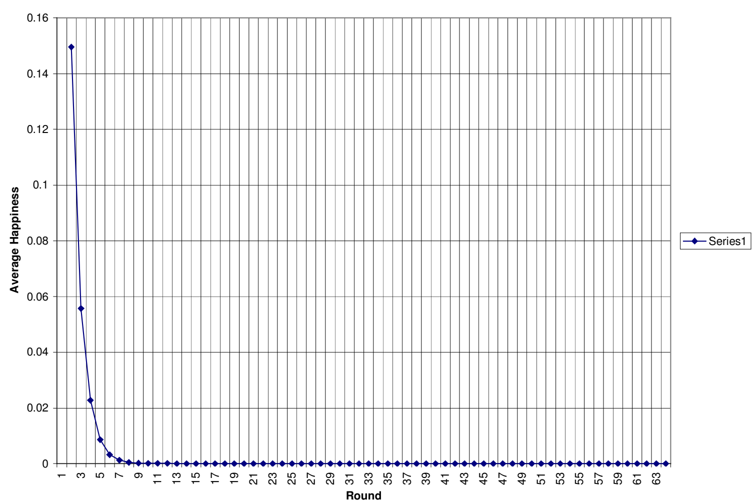
So far we have explored two baseline cases in which trading is not allowed. Now we will see what effect trade has on the behavior of a population of agents. 100 agents are created uniformly as follows:

Radius = 1
Trade Size = 0.5
Endowment = 7.0
Fairness = 0.5
Production Rate = 0.0

In this experiment, agents neither "produce" nor "consume" tokens: we set production/dissipation rate to zero in order to isolate the trading effect. Endowment is set to 7.0, so that average satisfaction begins around 1.5. This ensures a fairly wide distribution of satisfaction values across the population of agents, so that there is room, as it were, for trades to happen. Fairness is set at 0.5 to allow some "unfair" trading to occur.[11]

[11] Experiments have shown that when fairness is set at 1.0, very little if any trading occurs. I shall not stop to examine the more obvious social implications of this fact beyond saying that overly-strict short-term fairness stifles exchange.


Average Satisfaction
Round
Series1


Average Happiness
Round
Series1

Figures 4.3a-4.3c: Average satisfaction, average happiness, and Ω_S over time for a population of trading agents.

The results of this experiment are shown in figures 4.3a-4.3c. The effect of trading is quite pronounced. In the first few rounds, happiness is high as poorly endowed agents receive tokens from other agents who have extra tokens and are experiencing diminishing marginal returns. The population quickly equilibrates, so that by round 8 all agents have nearly equivalent $S \approx S_{avg} = 1.91$. After this point very few trades occur, and $H_{avg} \approx 0$. When the population is at equilibrium, tokens are distributed across the population so that total satisfaction is maximized. Figure 4.3d and 4.3e show the population before and after equilibrium has been reached.

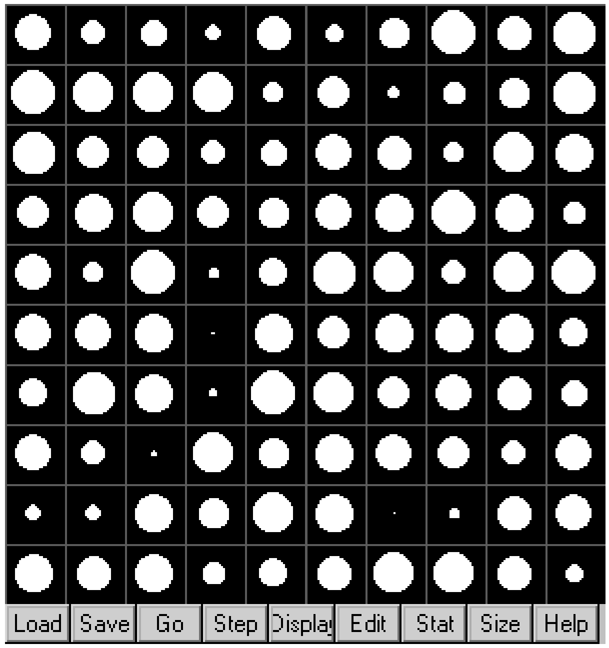


Figure 4.3d: Screenshot of the population in round 1 of experiment 3.

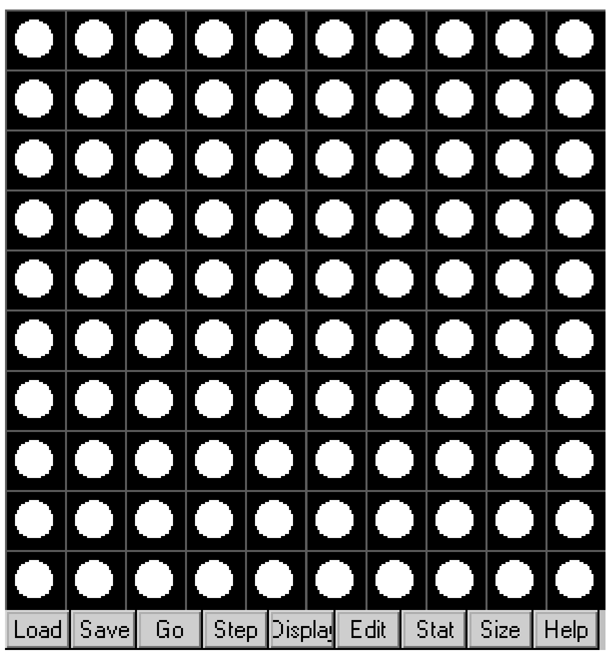


Figure 4.3e: Screenshot of the population in round 20 of experiment 3.

Since the result of trading is to equalize satisfaction values across the population, organization, R, increases with each round. Looking at figure 4.3c, we see that in the first round, where $C_S > R_S$, this increase in organization (and decrease in complexity) causes Ω_S to climb, until round 2 where $C_S \approx R_S$ and $\Omega_S \approx \Omega_{max} = 3.46$. After this point, $R_S > C_S$ and Ω_S falls to zero. H_{avg} is highest in round 2, which is the first round in which H_{avg} can be calculated since computing H_{avg} requires two S_{avg} values. Once again, this corresponds with the highest value of C_S (not counting round 1 since there is no H_{avg} value to compare). Without a graph of C_S, it is impossible to make the case for a directly proportional relationship between H_{avg} and C_S here, however such a relationship may indeed exist and should be explored in future work.

Chapter 5: Conclusions and Future Work

The experiments described in the previous chapter suggest that a direct correlation exists between the complexity of the frequency distribution of satisfaction values across a population of token trading agents and their average happiness over time. In all three experiments, C_S is greatest when satisfaction is changing the most, without regard to direction of change. This is not surprising, since C_S is simply a measure of the amount of variation in the S distribution, and S fluctuates the most when the magnitude of H_{avg} is large. Furthermore, our results show that when H_{avg} is accelerating, either upward or downward, Ω_S increases. Finally, we find that trade causes R and S to increase, moving the population toward an equilibrium state in which $R = R_{max}$, $C = 0$, $\Omega_S = 0$, $H = 0$ and S approaches a limit value.

While these experiments do establish a link between complexity and satisfaction, we feel that further experiments should focus on the complexity of *agent behavior* rather than the complexity of the satisfaction distribution. Future work should focus on one of three hypotheses, which we propose below. This will require a number of changes to the TokenTrade simulation, depending on which hypothesis we are trying to support.

HYPOTHESES 1 & 2

The first two hypotheses are opposite sides of the same coin. We hypothesize

> *that more complex-and-organized trading behavior always results in greater happiness,*

> and conversely,

> *that greater happiness is always the result of more complex-and-organized trading behavior.*

In order to run experiments with the TokenTrade simulation that will test these hypotheses, two things are necessary. First, a method of measuring the complexity-and-organization of agent behavior, and second, a collection of progressively "smarter" agents whose behavior is increasingly complex-and-organized. Then a series of experiments could be run to test if happier agents always exhibit greater "behavioral Ω" than their less happy counterparts, *and*, if all agents that exhibit complex-and-organized trading behavior are happier than their less complex-and-organized counterparts.

A simple method to measure complexity-and-organization of agent behavior would require keeping track of how often an agent trades with each of its surrounding neighbors. This would give a frequency distribution from which Ω could be calculated. For an agent with eight neighbors, we would compute C as follows:

$$C_B = -\sum p(1)\log_2 p(1) + p(2)\log_2 p(2) + p(3)\log_2 p(3) + p(4)\log_2 p(4)$$
$$+ p(5)\log_2 p(5) + p(6)\log_2 p(6) + p(7)\log_2 p(7) + p(8)\log_2 p(8)$$
$$+ p(0)\log_2 p(0)$$

....where $p(i)$ is the frequency of trade with neighbor i, and $p(0)$ is the frequency that the agent makes no trade at all.

Although this method would provide a good first approximation of behavioral complexity, it does not capture certain other information such as the amount and type of tokens traded. It also ignores temporal organization (sequential patterns) that may arise in an agent's trading behavior.

Once a method for determining behavioral complexity is established, agents must be created that exhibit increasingly complex-and-organized trading behavior. Currently, agent behavior is influenced by initial endowment, production rate, neighborhood size, maximum trade size and fairness. It may be possible, with only these parameters, to generate an array of different *types* of cells that exhibit sufficient variation in their trading behavior. However, experience with the TokenTrade simulation indicates that trading behavior is typically either extremely random or extremely ordered, and far from optimal Ω. Additional parameters governing trading behavior such as learning, memory, and "geographic" mobility would certainly allow more complex-and-organized behaviors to emerge.

Several sophistications that would increase the realism of the simulation and might increase the complexity-and-organization of agent behavior have been devised but not yet implemented. These sophistications are listed below:

Cost/benefit parameter(k). The cost/benefit parameter determines the degree to which giving tokens away costs or benefits the agent who is giving them. This could be individualized so that agent i has a different value k_j for each neighbor $j = 1,2,3,...,N$.

We compute the effect that giving away tokens has on satisfaction as follows:

$a = [k_j\ \Delta x]/[1+ k_j\ \Delta x]$,

$b = [k_j\ \Delta y]/[1+ k_j\ \Delta y]$,

$c = [k_j\ \Delta z]/[1+ k_j\ \Delta z]$,

where $\{\Delta x, \Delta y, \Delta z\}$ represents a token vector given to agent i by agent j, and where $-1 \leq k_j \leq 1$

$H = \Delta S = a(1 + b(1 + c))$

Implementing this change would require multiplying the `my_offer` vector by a parameter k_j before adding it to the `tokens` vector in both the `computeValue` and `exchangeTokens` methods.

Credence parameter(r). The credence parameter determines the effectiveness of tokens received from other agents. This could be individualized so that agent i has a different value r_j for each neighbor $j = 1,2,3,...,N$.

We compute the effect that receiving tokens has on satisfaction as follows:

$a = [r_j \Delta x]/[1+ r_j \Delta x]$,

$b = [r_j \Delta y]/[1+ r_j \Delta y]$,

$c = [r_j \Delta z]/[1+ r_j \Delta z]$,

where $\{\Delta x, \Delta y, \Delta z\}$ represents a token vector received from agent j by agent i, and where $-1 \leq r_j \leq 1$

$H = \Delta S = a(1 + b(1 + c))$

Implementing this change would require multiplying the `neighbor_offer` vector by a parameter r_j before adding it to the `tokens` vector in both the `computeValue` and `exchangeTokens` methods.

Enough parameter(E). $0.5 \leq a_E, b_E, c_E \leq 1$ is an agent's contentment level of each component of the satisfaction vector $[a,b,c]$, where $a = x/(1+x)$, $b = y/(1+y)$, and $c = z/(1+z)$, yielding a contentment vector $[a_E, b_E, c_E]$. When $a > a_E$, $b > b_E$, and $c > c_E$, the cell "retires" and stops trading. Implementing this change would require adding a check to see whether the retirement condition has been met in `findBestTrade`.

Value balance(L). $L_{i,j}$ is the value balance that cell i has with each of its neighbors $j = 1,2,3,...,N$. In other words, $L_{i,j}$ is the amount of happiness cell i owes cell j. The limit value for $L_{i,j}$ is $2r$. After this limit is reached, $L_{i,j}$ cannot be raised higher if $L_{i,j} > 0$. Adding value balance to the computation of V, we have:

$$V = H_{\text{cell}} - L_{\text{cell,neighbor}} + H_{\text{neighbor}} - L_{\text{neighbor,cell}}$$
$$- f \mid H_{\text{cell}} - L_{\text{cell,neighbor}} + H_{\text{neighbor}} - L_{\text{neighbor,cell}} \mid$$

HYPOTHESIS 3

The third hypothesis states

that the desire to increase happiness is the cause of complex-and-organized behavior.

The third hypothesis makes the evolutionary argument that in an environment of scarce resources, only those individuals that develop increasingly complex-and-organized behavior will be happy. We imagine a sort of bootstrapping process in which individuals find themselves in an increasingly complex environment as their neighbors evolve increasingly clever strategies for obtaining tokens. Individuals must adapt and refine their strategies in order to keep up, or else face the consequences. Those that can't keep up are removed from the gene pool. Over time, members of the population become highly adept at trading tokens, exhibiting increasingly complex-and-organized behavior.

This, of course, is the story of evolution applied to token trading. Although the idea that evolution trends toward increasing complexity-and-organization may seem self-evident, this is by no means an established scientific law. The prevailing scientific opinion is that it does, but empirical studies that would confirm this hypothesis remain to be done [McShea 1996]. Part of the difficulty is the controversy surrounding the concept of complexity and the lack of agreement on a single, suitable mathematical definition.

In order to test this third hypothesis, fundamental changes to the simulation would be required. Primarily, agents must be endowed with some way of adapting their behavior in response to their environment in order to improve their ability to acquire tokens. In essence, agents must *evolve*. This increases the programming challenge significantly, since evolving agents would necessitate some type of unsupervised machine learning algorithm. Furthermore, each agent would require its own independent "brain," and with a population of 100 or so agents, the learning algorithm would need to be sufficiently fast. Parallelizing the computations would be extremely beneficial as well.

One possibility would be to use genetic programming to breed a population of computer programs that learn to trade with one another. Genetic programming is a learning algorithm developed by Koza [1992] in which the evolutionary process is used to produce computer programs that perform a certain task. An initial population of programs is generated randomly, and each program is assigned a fitness score based on its success at solving the problem at hand. Successful programs survive and reproduce while unfit programs are replaced by the new offspring. Reproduction occurs though the genetic crossover of two parent programs, which results in a new program that contains part of the program of each parent. Over time, the population evolves and programs become increasingly adept at solving the specified task.

In genetic programming, one specifies a set of primitive functions and terminals (inputs) that serve as the building blocks from which programs are assembled. The function set typically includes operations such as *sum, product, divide, greater-than, if, abs, sin, cos, memory, min, max*, and so on. Let us imagine a population of programs where the task is trading tokens. For these purposes, we need to define two additional operations: *give* and *take,* both of which have two inputs, `neighbor` and `amount`. The *give* function gives the specified `amount` of tokens to the specified `neighbor`, while the *take* function takes the specified `amount` of tokens from the specified `neighbor`. Fitness scores would be determined by allowing the population of programs to run for a specified number of rounds, and then assigning each program a score based on its average happiness over time. The most successful token traders are chosen for reproduction and the process is repeated for a large number of generations. We would then compare the complexity-and-organization of successful and unsuccessful programs over the course of evolution to determine whether Ω increases along with happiness.

Although many details of this scheme remain to be worked out, we believe that such an experiment would provide invaluable insight into how complexity-and-organization arises in a population over the course of evolution, and would help us better understand the nature of the connection between evolution and complexity.

One final note. It is important to realize that hypothesis one (that more complex-and-organized trading behavior always results in greater happiness) and hypothesis three (that the desire to increase happiness is the cause of complex-and-organized behavior) are distinct, and that one does not necessarily follow from the other. It is quite possible that that hypothesis three is true but hypothesis one is false. Hypothesis one makes the strong claim that increased complexity-and-organization *always* results in greater happiness. This may not be true. In fact, it may be the case that complex-and-organized trading behavior correlates just as strongly with unhappiness as it does with happiness, just as it did with our pilot experiments. If hypothesis three were shown to be true in light of this, it would suggest that there is a world of complex-and-organized behaviors that simply never arise because they are not evolutionarily advantageous. Thus, the only way that we can say that increasing complexity-and-organization is *always* good for us is if we show all three hypotheses to be true.

References

Albin, P., and D. K. Foley. 1990. "Decentralized , Dispersed Exchange without an Auctioneer: A Simulation Study." *Journal of Economic Behavior and Organization* 18 (1): 27-51.

Arifovic, J. 1994. "Genetic Algorithm Learning and the Cobweb Model." *Jounal of Economic Dynamics and Control* 18: 3-28.

Arifovic, J. and C. Eaton. 1995. "Coordination via Genetic Learning." *Computational Economics* 8: 181-203.

Arthur, W.B. 1991. "Designing Economic Agents That Act Like Human Agents: A Behavioral Approach to Bounded Rationality." *American Economic Review, Papers and Proceedings* 81 (2): 353-59.

Arthur, W.B. 1994. "Inductive Reasoning and Bounded Rationality." *American Economic Review, Papers and Proceedings* 84 (2): 406-11.

Axelrod, R. 1993. "A Model of Emergence of New Political Actors." Working Paper 93-11-068. Santa Fe, N.M.: Santa Fe Institute.

Axelrod, R. 1995. "The Convergence and Stability of Cultures: Local Convergence and Global Polarization." Working Paper 95-03-028. Santa Fe, N.M.: Santa Fe Institute.

Axelrod, R. 1997. "Advancing the Art of Simulation in the Social Sciences." *Complexity* 3 (2): 16-22.

Benedikt, M. Forthcoming, 1999. *Value: Economics, Psychology, Life*. Chicago University Press.

Canning, A., and M. Droz. 1990. "A Comparison of Spin Exchange and Cellular Automata Models for Diffusion-Controlled Reactions." *Physica D* 45:285-92.

Carley, K. 1991. "A Theory of Group Stability." *American Sociological Review* 56: 331-54.

Danielson, P. 1992. *Artificial Morality: Virtuous Robots for Virtual Games*. New York: Routledge.

Danielson, P., ed. 1996. *Modeling Rationality, Morality, and Evolution*. New York: Oxford University Press.

Gardner, M. 1970. "Mathematical Games. The Fantastic Combinations of John Conway's New Solitaire Game "Life". *Scientific American* (October): 120-123.

Gardner, M. 1983. *Wheels, Life, and Other Mathematical Amusements*. San Francisco: Freeman.

Gilbert, N., and J. Doran, eds. 1994. *Simulating Societies: The Computer Simulation of Social Phenomena*. London: UCL Press.

Gilbert, N., and R. Conte, eds. 1995. *Artificial Societies: The Computer Simulation of Social Life*. London: UCL Press.

Hartman, H., and P. Tamayo. 1990. "Reversible Cellular Automata and Chemical Turbulence." *Physica D* 45: 293-306.

Holland, J., and J. Miller. 1991. "Artificial Adaptive Agents in Economic Theory." *American Economic Review, Papers and Proceedings* 81 (2): 365-70.

Kollman, K., J. H. Miller, and S. E. Page. 1992. "Adaptive Parties in Spatial Elections." *American Political Science Review* 86: 929-37.

Koza, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press.

Langton, C. G. 1990. "Life at the Edge of Chaos." *Artificial Life II*. Redwood City, Calif.: Addison-Wesley.

Li, W., N. H. Packard, and C. G. Langton. 1990. "Transition Phenomena in Cellular Automata Rule Space." Physica D 45: 77-94.

Marimon, R., E. McGrattan, and T. J. Sargent. 1990. "Money as a Medium of Exchange in an Economy with Artificially Intelligent Agents*." Journal of Economic Dynamics and Control* 14: 329-73.

Marks, R. E. 1992. "Breeding Hybrid Strategies: Optimal Behavior for Oligopolists." *Journal of Evolutionary Economics* 2: 17-38.

Maslow, A. [1954] 1970. *Motivation and Personality*. New York: Harper and Row.

McIntosh, H. V. 1990. "Wolfram's Class IV Automata and a Good Life." *Physica D* 45: 105-21.

McShea, D. W. 1996. "Metazoan Complexity and Evolution: Is There a Trend?" *Evolution* 50 (2): 477-92.

Nagel, K., and S. Rasmussen. 1994. "Traffic on the Edge of Chaos." Working Paper 94-06-032. Santa Fe, N.M.: Santa Fe Institute.

Seiburg, H., H. McCutchan, H., O. Clay, L. Caballero, and J. Ostlund. 1990. "Simulation of HIV-Infection in Artificial Immune Systems." *Physica D* 45: 208-28.

Shannon, C. E., and W. Weaver. 1949. *The Mathematical Theory of Communication*. Urbana: Univ. of Illinois Press.

Sites, J. 1995. "SFI's Next Generation of Models Tackles the Real World." *The Bulletin of the Santa Fe Institute*. Spring Bulletin. Santa Fe, N.M.: Santa Fe Institute.

Tesfatsion, L. 1995. "A Trade Network Game with Endogenous Partner Selection." *Economic Report* 36. Iowa State University, Department of Economics.

von Neumann, J. 1966. *Theory of Self-Reproducing Automata*. Edited by A. W. Burks. University of Illinois Press.

Vriend, N. 1995. "Self Organization of Markets: An Example of a Computational Approach." *Journal of Computational Economics* 8 (3): 205-31.

Wolfram, S. 1984. "Universality and Complexity in Cellular Automata." *Physica D* 10: 1-35.

Wolfram, S., ed. 1986. *Theory and Applications of Cellular Automata*. Singapore: World Scientific.

Vita

Robert Lavale Turknett, Jr. was born in Melbourne, Florida on October 21, 1971, the son of Robert Lavale Turknett, Sr. and Carolyn Norris Turknett. After completing his work at Lakeside High School in Atlanta, Georgia, in 1990, he entered Trinity University in San Antonio, Texas. During the fall of 1994 he attended the Institute for European Studies in Vienna, Austria. He received the degree of Bachelor of Science from Trinity University in December, 1994, where he published an article titled, “Artificial Evolution for Computer Music,” in the Proceedings of the National Conference on Undergraduate Research. After graduation he worked as a manager of strategic development at BellSouth Telecommunications in Atlanta, Georgia until September of 1995, when he entered The Graduate School at The University of Texas at Austin. He is currently employed as a software programmer/developer at Acuity, Inc., a software company in Austin, Texas.